# ORGANIZING, AND DOCUMENTING COMPONENT–ORIENTED TOOLKITS

By

MOHAMMAD AHMAD RADAIDEH, M.Sc.

A Thesis

Submitted to the School of Graduate Studies of McMaster University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

2000

McMaster University

# ORGANIZING, AND DOCUMENTING COMPONENT–ORIENTED TOOLKITS

**Doctor of Philosophy (2000)**
(Electrical and Computer Engineering)

**McMASTER UNIVERSITY**
Hamilton, Ontario, Canada

**TITLE:** Organizing, and Documenting Component–Oriented Toolkits

**AUTHOR:** Mohammad A Radaideh, M.Sc. (Jordan University of Science and Technology)

**SUPERVISORY COMMITTEE:**

Professor David Lorge Parnas (Supervisor)
Professor Ryszard Janicki
Professor Martin von Mohrenschildt
Professor Ridha Khedri

**EXAMINATION COMMITTEE:**

| | |
|---|---|
| Professor D. Shaw | (Chair) |
| Professor D. Schmidt | (External Examiner) |
| Professor H. Stover | (Examiner – McMaster) |
| Professor A. Hitchcock | (Examiner – McMaster) |
| Professor D. Parnas | (Supervisor) |
| Professor M. Mohrenschildt | (Supervisory Committee Member) |
| Professor R. Khedri | (Supervisory Committee Member) |

**NUMBER OF PAGES:** xiii + 407 pages

# ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor Professor David Parnas for his continuous support and guidance since September 1995. I would like to thank as well, Professor Ryszard Janiscki, Dr. Martin von Mohrenschildt, and Dr. Ridha Khedri for serving on my supervisory committee. Also, I would like to Professors Shaw, Stover, and Hitchcock from McMaster university for serving on my examining committee. A special thank you goes to Dr. Douglas Schmidt for the detailed comments he provided in his report as the external examiner for my thesis.

I gratefully acknowledge the financial support provided by the Telecommunication Research Institute of Ontario (TRIO), and the Department of Electrical and Computer Engineering at McMaster University during the first two years of my study at McMaster. Special thank you goes to Michael White from IBM Canada for his continuous encouragement and support to complete this thesis.

My parent, brothers, and sisters deserve much more than "thank you" for their sincere encouragement.

Finally, I would like to express my grateful appreciation and special thanks to my wife Rawiah, and our beloved sons Sa'eb, Ahmad, Saleh, and Khaled for their continuous patience, love, encouragement, and moral support throughout these long years.

# ABSTRACT

Component–Oriented Software Technology started to emerge during the last few years. The purpose is to make it easier to build new applications from existing components that can be obtained from Component–Oriented Toolkits (CO–Toolkits for short). Experience shows that CO–Toolkits are hard to understand and use. Therefore, developers would write their own code from scratch using programming languages that they know instead of using components from such CO–Toolkits.

In this research, we have used ADAPTIVE Communication Environment (ACE for short) – as an example of CO–Toolkits – to establish a set of initial assumptions for good design practice from the perspective of their file structures, design and implementation, and documentation of CO–Toolkits. Then (i) we show how to formalize rules for good file structures, design and implementation, and documentation that are based on these initial assumptions, (ii) we show that checking tools can be built based on these formalized rules, and then (iii) we demonstrate that such tools can be used to check existing CO–Toolkits code and report any rule violations back to developers.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Overview

Component–Oriented toolkits (i.e. CO–Toolkits for short) started to emerge during the last few years. CO–Toolkits provide developers with components that can be used to build new applications. This can save developers a lot of time since they do not have to write their applications from scratch.

The main objectives of this research are: (i) to show how to formalize rules and guidelines for good design practice, (ii) to show that such formalized rules can be used to develop tools, and (iii) to show that such tools can be used to check CO–Toolkits' code against these formalized rules. In order to demonstrate the practical value of these results, and for the reasons mentioned in the next chapter (*c.f. section 2.2.2*), we chose the "ADAPTIVE Communication Environment" (i.e. ACE for short) toolkit as a case–study for this research [41, 1, 43–52].

Many words (i.e. frameworks, component, etc.) are commonly used by software developers with inconsistent meanings. In the following paragraphs, we show how such words are used throughout this thesis.

In [31], Meyer stated that "*there is no generally accepted definition for components*". However, he described a component as "a *program element with the following properties: the element may be used by other program elements (client), and the clients and their authors do not need to be known to the element's author*". In [2], Nelson referred to components as "*classes or frameworks*". In [19], Fayad and Schmidt defined components as "*self–defined instances of abstract data types (ADTs) that can be plugged together to form complete applications*". In [28], Lakos defined a component as follows: "a *component is the smallest unit of physical design*".

As there is no generally accepted definition for components, a component (i.e. throughout this thesis) is a pre–defined application that is composed of a set of internal classes, and a main part such that (i) an internal class is a class in the context of object–oriented languages, and (ii) a main part is a program that contains the component's execution entry point (i.e. main function). Also, a main part can contain other functions that do not belong to any of the component's internal classes.

For convenience, we used three files for each component. We refer to them as the top, intermediate, and internal files. The top file is intended to hold the component's main part, the intermediate one is intended to hold the component's interface, and the internal one is intended to hold the data structure of the component as well as the implementation of all of its internal functions and methods.

According to Schmidt [43], the "component collaboration" can be defined as follows: "a set of components is considered to be collaborating if they exchange services".

The "component invocation" is used throughout this thesis in the sense that a component is considered invoked by an application when that component is executed at run time from within that application. In such a case, the user's application needs only to have a proper path to the directory under which the invoked component is stored. Also, it is considered invoked when its classes are "inherited from" and/or "associated with" by the application. In this case, the application needs to include the files used to implement the invoked component.

According to Schmidt [43], a "framework" is defined as follows: "*A framework is an integrated collection of components that collaborate to produce a reusable architecture for a family of related applications*". For example, a set of components that provide signal detection, signal sampling, signal data manipulation and processing, etc. can be considered as a single framework that provide a set of relevant signal processing services.

We define a "CO–Toolkit" as a toolkit that provides developers with components. Inside a CO–Toolkit, components are grouped into a set of frameworks. In order to keep frameworks within the same CO–Toolkit independent from each other, shared–pools are suggested such that each shared–pool contains the set of classes and other entities that are shared by a number of the frameworks.

The word "good" is used throughout this thesis in the sense that a "good" thing is a thing that can be "useful" and "beneficial". For example, we have discussed in chapter 3 the assumptions for "good" design practice. We think that these assumptions can be useful if they are considered in the construction of CO–Toolkits.

The word "perspective" is used throughout this thesis in the sense that a software perspective can be described in a formal relation. For example, file structures, implementation, and documentation of CO–Toolkits are three separate perspectives of CO–Toolkits.

The word "view" is used throughout this thesis in the sense that a "view" is an abstraction for reducing complexity of CO–Toolkits from a particular perspective through a formal structure and a set of rules and guidelines. Others used the word "view" for meanings that are close to our meaning for the word "view". For example, Embley [17] defined the word "view" as *an abstraction mechanism for reducing complexity in large object_oriented system analysis models.* Others like Ayers [3], Goldberg [22], and Rumbaugh [42] defined the word "view" as *the method of presenting the information contained in the underlying model of an application.*

The word "concern" is used throughout this thesis in the sense that a software concern is a matter or a design decision that needs the attention of the software developer who should be able to modify that concern without having to worry about the detail of the software's other concerns. According to the Webster's dictionary [29], a concern is defined as "*a matter that engages a person's attention, interest, or care*".

We have looked at ACE –as an example– from the perspectives of its file structure, relations (i.e. inheritance and association) among its components, and its documents (*c.f. chapter 2*). Based on (i) what we learned from ACE, (ii) software design principles as explained in the first half of chapter 3, and (iii) object–oriented programming experience gained during the last several years, we made a set of assumptions that can be good for software design practice (*c.f. chapter 3*).

Based on the assumptions referenced in the previous paragraph, we derived and formalized a set of rules for: (i) how to organize CO–Toolkits' file structures (*c.f. chapter*

*4*), (ii) how to implement CO–Toolkits' components (*c.f. chapter 5*), and (iii) how to document CO–Toolkits (*c.f. chapter 6*).

Based on the formalized rules (i.e. views) referenced in the previous paragraph, we developed prototype systems that can be used to check CO–Toolkits' file structures (*c.f. Appendix A*), and their implementation structures (*c.f Appendix B*) against these formalized rules. These prototypes report any violation to any of these formalized rules back to developers (*c.f. Appendixes A2–A4, B2–B4*). Java has been used to write these prototypes because we are familiar with it.

The formal models presented in this thesis (*c.f. chapters 4–6*), and the prototypes (*c.f. appendices A and B*) are intended to be general, but they are incomplete. The "Future Work" section (*c.f. chapter 8, section 8.4*) describes further potential improvements on the rules and the prototypes. Our prototypes were intended just to show that formalized rules can be used to develop checking tools. Therefore, we think that it is not necessary to provide any front ends that can be used to directly access CO–Toolkits' source code.

As shown in Figure 1–1 next, we expect a separate front end to extract the necessary artifacts from the CO–Toolkit's source code and build the various relations (i.e. the file structure, implementation, documentation, and consistency relations) that are required by the prototypes in order to check each for its own rules. Given that the purpose of our prototypes is just to show that formalized rules can be used to build tools, we have chosen to develop prototypes only for the first two views. Building prototypes for the third view and the views' consistency is left for future work (*c.f. chapter 8, section 8.4:(4)*).

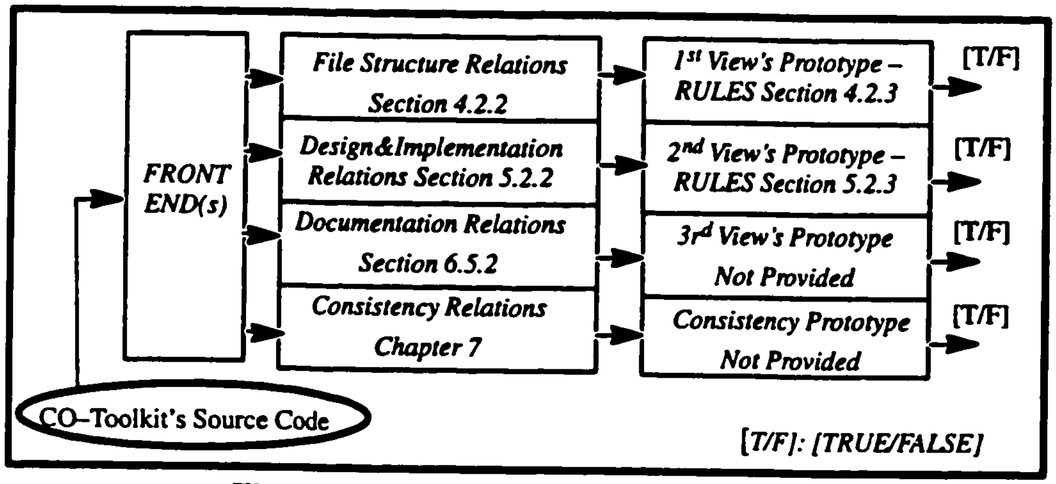


**Figure 1-1: CO–Toolkits' Checking Cycle**

This chapter has seven sections. The second section describes the motivation for this research. The third section describes some of the related work. The fourth section describes the audience intended by this research. The fifth section describes the primary common characteristics of the CO–Toolkits. The sixth section describes the main contributions made by this research. The last section describes how this thesis is organized.

## 1.2 Research Motivation

The motivation for this research can be summarized as follows:

(1) Upon starting this research, the guiding principle was "before using components from a given CO–Toolkit, developers need to understand that toolkit first, and to understand it, they need proper documents".

(2) CO–Toolkits provide components that can be used to build new applications. This is intended to simplify, enhance, and speedup the process of software development. However, since proper documents are not available with such systems, users of CO–Toolkits find it hard to understand and use their components. As a result, the process of software development slows down.

(3) Precise documents can help to better understand CO–Toolkits. Such documents can save developers a lot of time. As a result, the process of software development becomes faster. For example, ACE documentation comes in a set of academic papers [43–52] that we think were intended for people who are already familiar with ACE. Users can not just rely on these papers to understand how to use components from ACE.

(4) The need for precise documents encouraged us to investigate how to produce them. To be able to produce them, CO–Toolkits should be well–organized and designed.

(5) The need for checking tools that can be used to check existing software systems against a certain set of rules and guidelines and then report problems back to developers. We think that formalizing rules can make it easier to build such checking tools.

## 1.3 Related Work

### 1.3.1 The "Automatic Detection of Design Problems" Work

#### 1.3.1.1 What Is The "Automatic Detection of Design Problems"?

This work is being performed at the FZI Forschungszentrum Informatik and it has been introduced to us around the time of completing this thesis. All details provided in this section are based on a paper [14] by Dr. Ciupke from the above mentioned organization.

This work has been focused on analyzing legacy code (i.e. non object–oriented code), simplifying frequent design problems as queries and locating occurrences of these problems in a model derived from source code. A tool set has been implemented to support these tasks. The following are some of the criteria that have been considered in this work:

(1) Methods and tools work on the design level, and the quality of the overall structure of a system, determines the flexibility of that system.

(2) All of the required information is extracted from the source code of the input system.

(3) To cope with large amounts of code, problem detections should run automatically.

(4) A problem detection method should point us directly to the problems we are searching for.

(5) The approach should be language independent.

(6) An approach to problem detection in reengineering should cover a wide range of possible problems.

#### 1.3.1.2 Our Work vs. The "Automatic Detection of Design Problems" Work

"Automatic Detection of Design Problems" approach is relevant to our work in the sense that:

(1) Similar to their approach, our approach is based on identifying a well–defined set of problems that can be detected based on a set of well–defined rules.

(2) Similar to their tool set, our prototypes can be used to check source code against pre–defined set of rules and guidelines. Any violations to these rules will be reported back to developers.

(3) Similar to their methods, our methods (i.e. rules) are language–independent.

(4) The main difference between the "Automatic Detection of Design Problems" (i.e. ADDP for short) and our research, is that the ADDP is focused on legacy software systems, while our approach has been focused on Component–Oriented Toolkits.

## 1.3.2 Rigi

### 1.3.2.1 What is Rigi?

As described in [57], Rigi is a system used to understand large information spaces like software programs. Rigi can model the input system by extracting artifacts from the information space using a language–dependant parser (e.g. cparse for C programming language), organizing them into higher level abstractions, and then visualize them using a graph editor called rigiedit.

The rigiedit is the user interface that is used to browse, analyze, and modify a graph that models the input system. The rigiedit can be used to simplify the graph by clustering related artifacts into subsystems. This clustering is mainly based on the components functions.

The rigiedit has built–in operations to assist in program understanding. It can be used to select and group artifacts based on certain modularity principles such as data abstraction within subsystems. Various statistical reports can help with maintenance or re–engineering tasks. Also, rigiedit is programmable using a script language called Tcl [35].

### 1.3.2.2 Our Prototypes vs. Rigi

Rigi is mainly a visualizing tool that can be used to visualize artifacts that are extracted from code using other parsing tools.

Similar to Rigi, our prototypes are dependent on appropriate parsers that can be used to extract the required artifacts (i.e. #include statements, inheritance, association relations, etc.) from the source code. What makes our prototypes different from Rigi is that they are checking engines that check code against certain set of formalized rules and guidelines (*c.f. chapters 4 and 5*). These formalized rules are based on the initial assumptions for good design practice that we made in chapter 3.

### 1.3.3 Together/C++

#### 1.3.3.1 What is Together/C++?

Object International [NO TAG, 34] introduced the Together/C++ visual development tool. Source code (i.e. C++ code) can be developed (or imported in) and visualized using the Together/C++ tool. Unlike Rigi, Together/C++ tool uses its own embedded parsing utility and keeps its own internal repositories to hold the artifacts.

#### 1.3.3.2 Our Prototypes vs. Together/C++

Our prototypes are not development tools like Together/C++. They can be used to check the input source code against a certain set of rules and guidelines (*c.f. chapters 3 and 4*). These rules and guidelines can be described as an extension to programming languages in the sense that existing programming languages may permit many situations where our rules can be violated. While the Together/C++ tool permits any situation that is permitted by C++.

## 1.3.4 DATRIX

### 1.3.4.1 What is DATRIX?

DATRIX is a parsing tool developed at Bell Canada. It can be used to extract many kinds of artifacts from the source code such as the #include relation, data flow, etc. GRAPHPLACE CASE is then used to postscript these extracted artifacts and then tools like GHOSTVIEW can be used to visualize them [8, 9].

### 1.3.4.2 Our Prototypes vs. DATRIX

DATRIX can be related to our work in the sense that it can be modified such that it can then be used to extract the artifacts that are required by our prototypes to conduct their checking analysis.

## 1.3.5 Other Parsing Tools

There are many parsing tools available from both the academic institutions and the industrial organizations (i.e. software development houses). We can group parsing tools into two main groups:

(1) *Programming language–specific parsers:* This group contains any parser that is used by a language–specific compiler to parse in source code that needs to be compiled. Also, it contains any parser that is used by a language–specific linker to parse in object code that needs to be bound.

(2) Task–specific parsers: This group contains any parser that is used to extract particular set of artifacts from its input source.

#### 1.3.5.1 Our Prototypes vs. Parsing–Tools

Given that our prototypes check for a particular set of rules and guidelines that we introduced in this research, only the language task–specific group of parsing tools can be complementary to our prototypes.

A task–specific parser can be integrated into our prototypes if that parser can extract the set of artifacts from the given language–specific source code such as the "#include" statements, the inheritance and association relations, objects (and their types) etc.

## 1.4 Our Audience

This research is intended for the following people:

(1) Developers of CO–Toolkits:

*i–* We think that following our guidelines in documenting, designing and organizing CO–Toolkits will make it easier to maintain such toolkits.

*ii–* Developers can use tools that are developed based on our views to check their CO–Toolkits' code.

(2) Maintainers and technical support staff of CO–Toolkits:

*i–* If developers provide precise documents with their CO–Toolkits, maintainers will be able to find the parts of their systems' code that need to be modified in order to fix bugs or change some features.

*ii–* However, it is the maintainers obligation to keep such documents always up–to–date, otherwise, such documents will loose their preciseness and correctness.

(3) CO–applications developers (i.e. "Users" of CO–Toolkits):

*i*– Users can use our documents to determine if the given CO–Toolkit has particular components that can provide their applications with certain services.

(4) Software testers/evaluators of CO–Toolkits and their applications can use our documents to understand the components they are dealing with.

(5) Given that a Component Guide Document (i.e. CGD for short) that we propose in chapter 6 can contain graphical pictures about its system, sales staff can use some of these pictures for their advertisements.

## 1.5 The Primary Common Characteristics of CO–Toolkits

The common characteristics of CO–Toolkits that are the focus of our research, could be summarized as follows:

(1) A CO–Toolkit is a software system that is composed of a set of frameworks such that each framework is composed of a set of components.

(2) A CO–Toolkit is developed using an object–oriented programming language such as C++, etc. It may be possible to develop CO–Toolkits using non Object Oriented Languages, however, the outcome of this research can not be used for them.

(3) Integration and Composability: A component must be able to integrate with one or more other components from the same system to build a new application.

(4) Basic concepts of software engineering such as information hiding, design for change, expandablity, etc.[21] must be maintained in such CO–Toolkits:

*i*– Information hiding and encapsulation (i.e. separating the interface from the implementation): The interface is a contract that specifies a component's functionality. The interface should not expose any implementation or platform or environment dependencies.

*ii–* Expandablity of CO–Toolkits means that developers and maintainers of such toolkits can add new components to any framework without having to change any of its existing components. Also, they should be able to add new frameworks without having to change any of the existing ones.

## 1.6 Key Contributions Made By This Research

The main contributions made by this research can be summarized as follows:

(1) We have shown how to formalize rules for good practice concerning file structures (*c.f. chapter 4*), design and implementation (*c.f. chapter 5*), and documentation (*c.f. chapter 6*) of CO–Toolkits. This is a contribution because formalization of rules is an important step towards building tools. As explained next, we found it easy to build our prototypes based on the formalized rules made in this thesis.

(2) We have shown that such formalized rules can be used to develop tools. In this thesis, we provided prototype systems (*c.f. optional reading material is provided in appendixes A and B*) that we developed based on these formalized rules (i.e. views).

(3) We have shown that such prototypes can be used to check software systems and report any rules violations back to developers (*c.f. Optional reading material is provided in appendixes A2–A4, B2–B4*).

## 1.7 Thesis Organization

This thesis has eight chapters, and three appendixes:

- Chapter 1 is this introduction chapter.
- Chapter 2 describes the case study of this research (i.e. ACE).
- Chapter 3 describes the set of assumptions for good design practice that we made based on our experience with ACE as well as based on our own work experience. Based on these assumptions, we derived and formalized rules as described in the three views of this research (*c.f. chapters 4 through 6*).
- Chapter 4 describes our view for CO–Toolkits' file structures.
- Chapter 5 describes our view for CO–Toolkits' design and implementation structures.
- Chapter 6 describes our view for CO–Toolkits' documentation.
- Chapter 7 shows how to check three views to be sure that they are consistent.
- Chapter 8 describes the limitation of our views, conclusions, and future work.
- Appendix A describes the prototype that we built based on the view for CO–Toolkits' file structure of chapter 4. Appendix A1 describes how to use this prototype as well as describes and presents its source code. Appendix A2 presents the first artificial example that has been used to demonstrate the kind of file structures that can be passed by the prototype. Appendix A3 presents the second artificial example that has been used to demonstrate the kind of file structures that will be failed by the prototype, and to demonstrate the kind of violation messages that are reported back to developers. Appendix A4 presents a third example that is based on a sample ACE file structure.
- Appendix B describes the prototype that we built based on the view for CO–Toolkits' implementation structures of chapter 5. Appendix B1 describes how to use this prototype as well as describes and presents its source code. Appendix B2 pres-

ents the first artificial example that has been used to demonstrate the kind of design and implementation structures that can be passed by the prototype. Appendix B3 presents the second artificial example that has been used to demonstrate the kind of design and implementation structures that will be failed by the prototype, and to demonstrate the kind of violation messages that are reported back to developers. Appendix B4 presents a third example that is based on a sample ACE inheritance structure.

- Appendix C provides a set of sample tables for the view for CO–Toolkits' documentation of chapter 6.

# CHAPTER 2

# CASE STUDY: STRUCTURE AND ANALYSIS

## 2.1 Overview

This chapter describes the "ADAPTIVE Communication Environment" (i.e. ACE for short) that has been used as a starter example for this research. The ACE version used for this research goes back to April 1996. At that time ACE was suggested as a good example to look at by people from the industry at a TRIO meeting. Improvements may have been made on ACE since that time, however, this research was not intended to keep up with such improvements. As explained in the next chapter, after looking at ACE, we looked back at the software engineering principles and then derived a set of assumptions for good design practice.

We think that a less coupled software design (i.e. designing software for change and mastering complexity) would reduce the learning curve in any software area. However, someone may argue that this can be true in some areas such as GUI frameworks, but not necessarily true in some other areas such as telecommunications, medical imaging, or real–time embedded systems where efficiency, predictability, and scalability are key forces. We think that "designing software for change and mastering complexity" needs to be always a key force in order to have systems that can be maintained at reasonable costs. It may be hard to achieve a software system that is designed for a high degree of performance and a high degree of simplicity at the same time. However, with the "designing software for change and mastering complexity" as the key force, software systems can be constructed or reconstructed in away such that performance implications can be minimized. For example, providing precise and correct specifications for the design and the interface of a CO–Toolkit can help developers in choosing the right components for their applications.

This chapter has six sections. The next section describes ACE. In the third section, we discuss the need for studying CO–Toolkits' file structures, and present samples of the file structures that we derived from ACE frameworks. In the fourth section, we discuss the need for studying the inheritance and association relations among CO–Toolkits' components, and present samples of these two relations as used among ACE components. The fifth section describes the kinds of documents that we found with ACE and the need for more documents. The last section presents the conclusions of this chapter.

## 2.2 Case Study: ADAPTIVE Communication Environment

This section describes ACE according to Schmidt in his papers on ACE [43–52].

### 2.2.1 What Is The "ADAPTIVE Communication Environment"?

The ADAPTIVE Communication Environment is a freely available object–oriented toolkit that provides a rich set of reusable components that can be used to perform common communication software tasks across a range of platforms. The communication software tasks provided by ACE include event demultiplexing and event handler dispatching, signal handling, service initialization, inter–process communication, shared memory management, message routing, dynamic configuration of distributed services, concurrent execution and synchronization. Many benefits can be accomplished by using ACE. Examples are: (i) increased portability, (ii) increased software quality, and (iii) increased efficiency and predictability.

The "ACE OS Adapter Layer" resides directly above the Operating System (i.e. OS for short) APIs (i.e. Application Programming Interfaces) written in C. It provides platform–specific components that are associated with the following OS APIs: (i) concurrency and synchronization, (ii) inter–process communication (i.e. IPC for short) and shared memory, (iii) event demultiplexing mechanisms, (iv) explicit dynamic linking, and (v) file system mechanisms.

The "ACE C++ Wrapper for OS Interfaces" simplifies application development by providing C++ interfaces that encapsulate the OS concurrency, communication, memory management, event demultiplexing, and others. Applications can use from these wrappers (i.e. programs) by, for example, inheriting and/or instantiating the following components: (i) concurrency and synchronization components, (ii) IPC and file system components, and (iii) memory management components.

The "ACE Frameworks Layer" is a higher–level programming layer that integrates the lower–level C++ wrapper (i.e. programs). It contains the following: (i) Event Demultiplexing components, (ii) Service initialization components, (iii) service configuration components, and (iv) hierarchically–layered stream components.

### 2.2.2 Why Did We Choose ACE As A Case Study?

We have selected ACE as a case study for the following reasons:

(1) It was presented by industry to the Telecommunication Research Institute of Ontario (i.e. TRIO for short) as an example of something that they could use.

(2) ACE provides a rich set of components (over 300 classes categorized into several frameworks). Thus, it would serve as a good case study for this research.

(3) It has been developed to meet the high demand for robust and high–performance distributed communication software systems such as global communication systems, network management platforms, etc.

(4) It has been used to build real applications in the fields just mentioned before.

(5) Its code is available for free and the author (Dr. D. Schmidt) has agreed to help.

### 2.2.3 Are Our Views ACE–Dependent?

As we think that ACE is a good representative of CO–Toolkits, our mathematical views (*c.f. chapters 4–6*) are intended to be general such that they can be used for any CO–Toolkit that satisfies the characteristics stated in section 1.5 of chapter 1.

## 2.3 File Structures

### 2.3.1 Why Do We Need To Study File Structures?

Software developers reuse files among the components of their systems without paying enough attention to the growing dependency among these files. As a result, the "#include" relation among files (i.e. file dependencies) becomes complex and unorganized [9]. This has encouraged us to investigate how CO–Toolkits' file structures should be organized where the "designing software for change and for mastering complexity" is the key force.

## 2.3.2 Visualizing ACE File Structure

### 2.3.2.1 ACE File Structures Graphs

This section describes snapshots of the existing ACE file structure. We used DATRIX and GRAPHPLACE CASE tools [8] to extract and visualize the "#include" statements from ACE files as shown in Figures 2–1 through 2–6.

As ACE has been used as a research subject, the following examples from ACE are presented in this section for the purpose of demonstrating our ideas. For example, pointing to the bi–directional inclusion relation between the ACE.h and ACE.i files (i.e. in Figure 2–4) is intended to explain what we mean by bi–directional inclusion between the intermediate and the internal files of the same component.

Figure 2–1 shows the file structure of all ACE components is not easy to understand. This would be the result of using an add–hoc approach during the development process of software systems. When it comes to adding new classes to an existing system, developers try to reuse as much of the existing entities (variables, classes, etc.) as possible. They simply "#include" the files that contain these entities without watching the growth of their software systems' file structures.

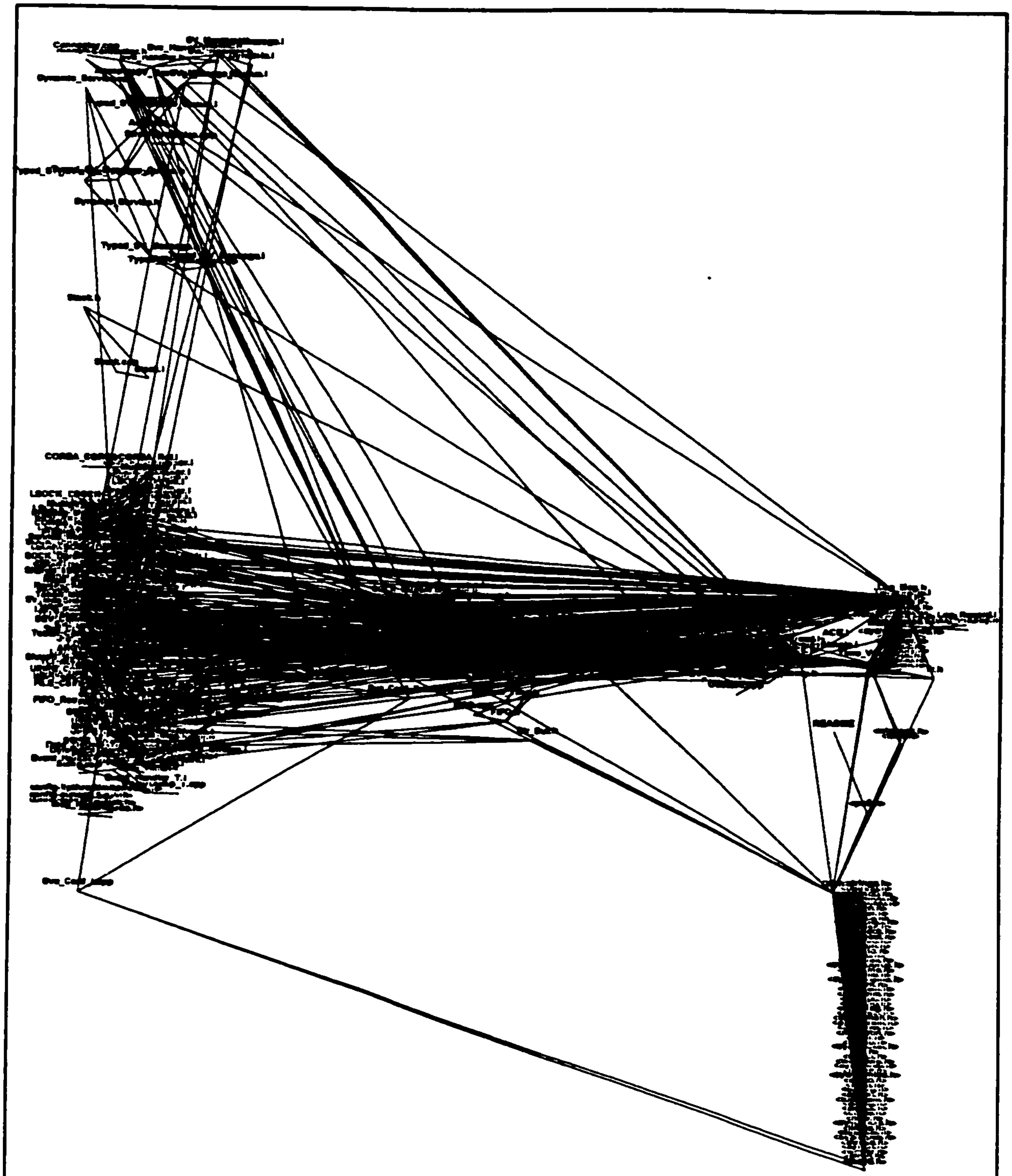

**FIGURE 2–1 :** Overall ACE File structure

Figure 2–2 shows the file structure of the REACTOR framework of ACE. This graph comes with four extension nodes that are illustrated in Figures 2–3 through 2–6. Each extension node is intended to represent a pattern sub–graph that is used in several places (i.e. OS.h). As shown in Figures 2–2 through 2–4, each extension node is denoted by "++++++included–file++++++".

Figure 2–2 shows the following multiple path case: Event_Handler_T.cpp "#include"s both Event_Handler_T.h and Event_Handler_T.i, and Event_Handler_T.h "#include"s Event_Handler_T.i. It is clear that Event_Handler_T.cpp has two possible paths to "#include" Event_Handler_T.i, either directly or through including Event_Handler_T.h. Many other multiple–path cases can be noticed in Figures 2–2 and 2–3.

Figure 2–4 shows a sequence of inclusions in which Log_Msg.h header file "#include"s Log_Record.h, Log_Record.h "#include"s ACE.h, then ACE.h "#include"s ACE.i, which in turn "#include"s Log_Msg.h forming a loop. In this graph we notice two things: having loops, and having a .i file "#include"ing a .h file.

### 2.3.2.2 What Issues Do These Figures Raise?

Figures 2–2 through 2–6 raise issues such as having inclusion loops, and inclusion multiple paths in file structures. We understand that a language like C++ [26] can handle these issues through special checking statements that precede the "#include" ones. Such checking statements ignore any "#include" statement if the file to be "#included"d has been "#include"d before in the current sequence of inclusions.

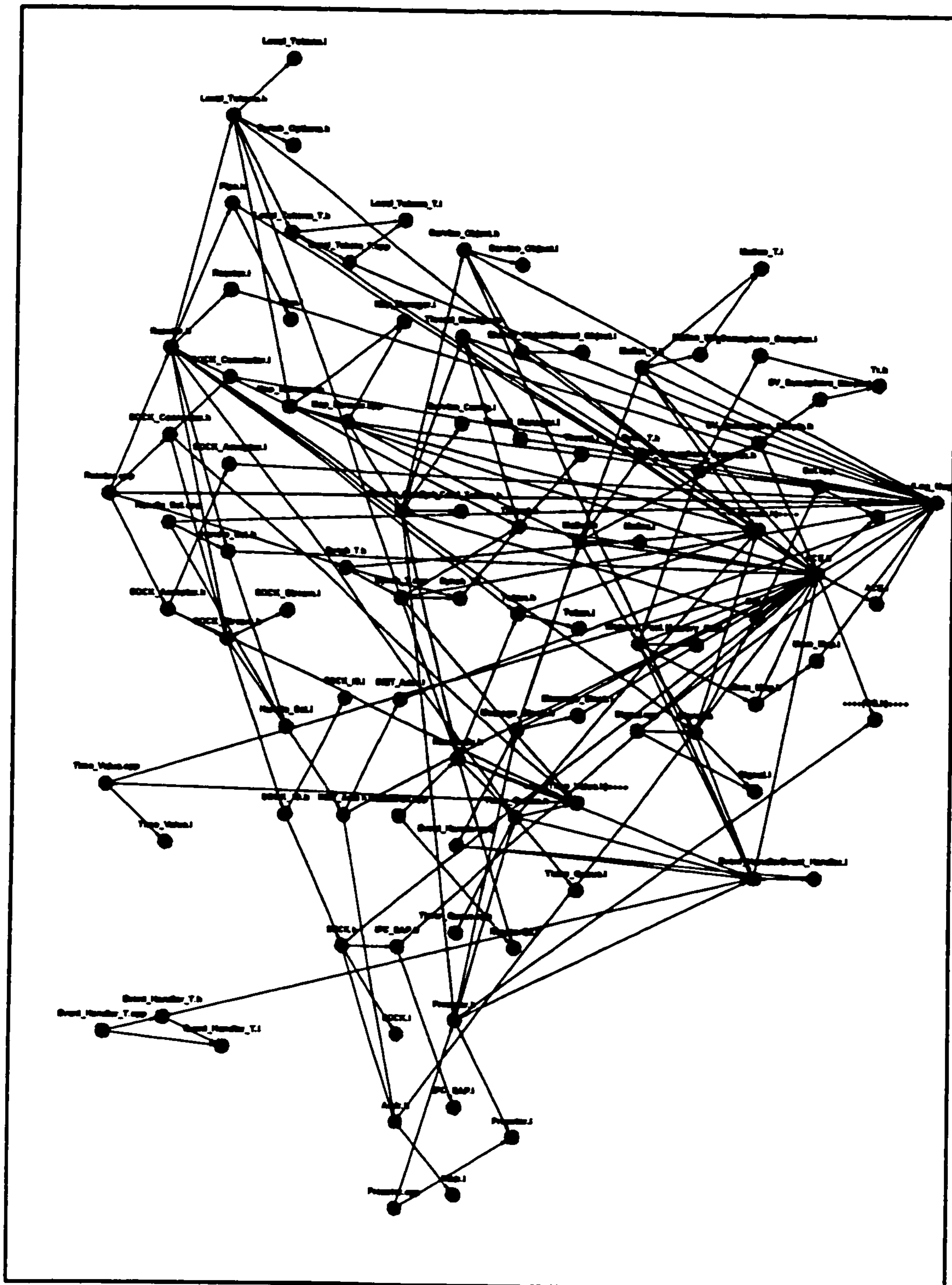

**FIGURE 2–2 :** REACTOR Framework File Structure

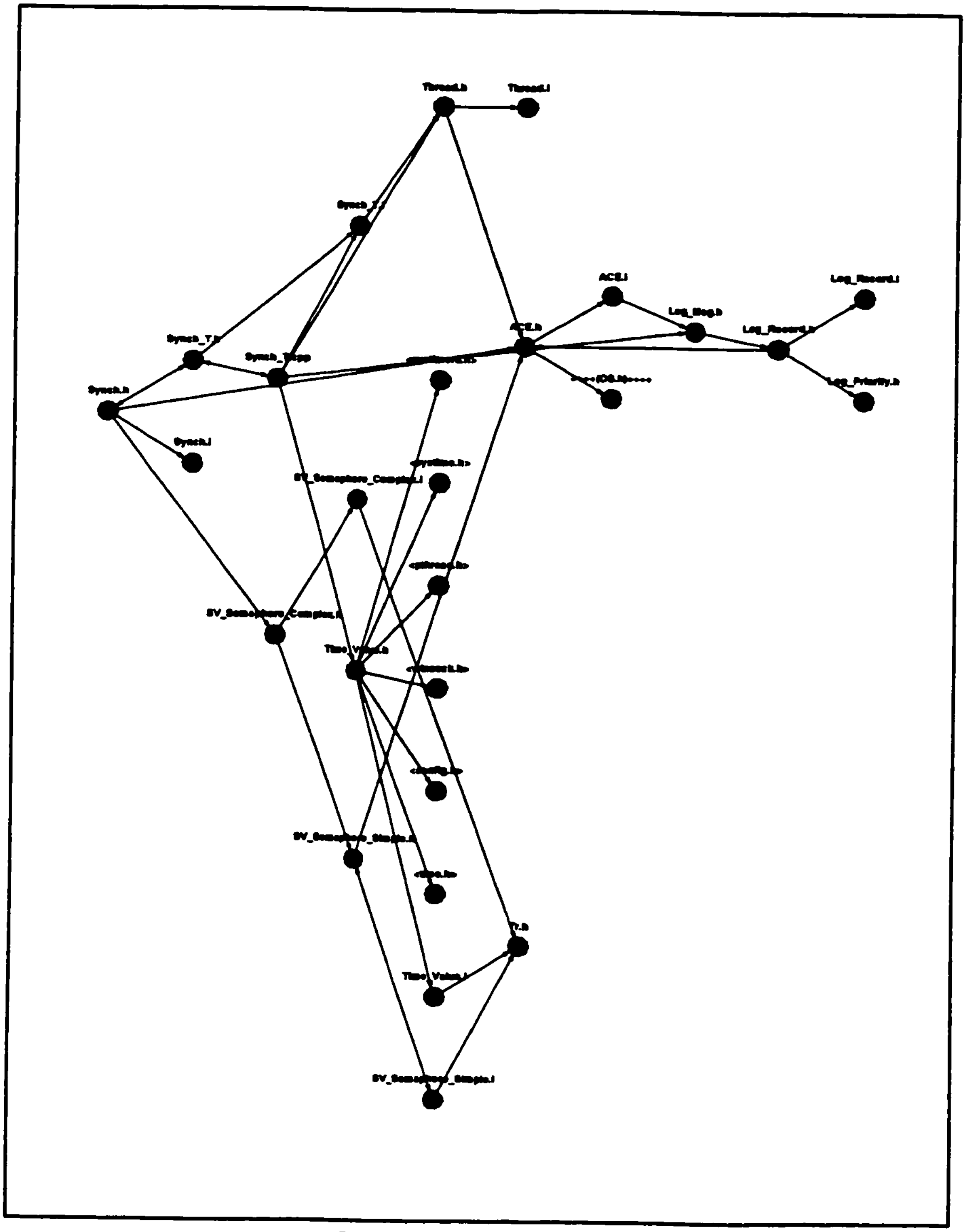


**FIGURE 2–3 :** Synch.h File

Having a software system to pass compilation and then run does not necessarily mean the system has been well-organized. A File of a well-organized system need to "#include" other files only when there is a need to do so.

### 2.3.2.2.1 Inclusion Loops

We define an inclusion loop as a sequence of files such that each file "#include"s its next in the sequence, and the last file "#include"s the first one in the same sequence. In other words, an inclusion loop could be viewed as a ring of file inclusion operations.

As a special case of inclusion loops, it is interesting to mention that each of the bi-directional inclusions in ACE exists between a pair of files that share the same file-name with different extensions (i.e. one with .h and the other with .i extension.). Given that an internal file is intended to contain its component's internal data structure and the top file is intended to host the user interface, we think that such bi-directional inclusions violate the software design concept of information hiding [36–40].

### 2.3.2.2.2 Inclusion Multiple Paths

We mean by multiple-paths that a file could "#include" another file through more than one possible sequence of inclusions. For example; let file A "#include"s files B and C, file B "#include"s files D and E, and file C "#include"s files E and F. In this example, file A "#include"s file E through two possible sequences; <A, B, E> or <A, C, E>.

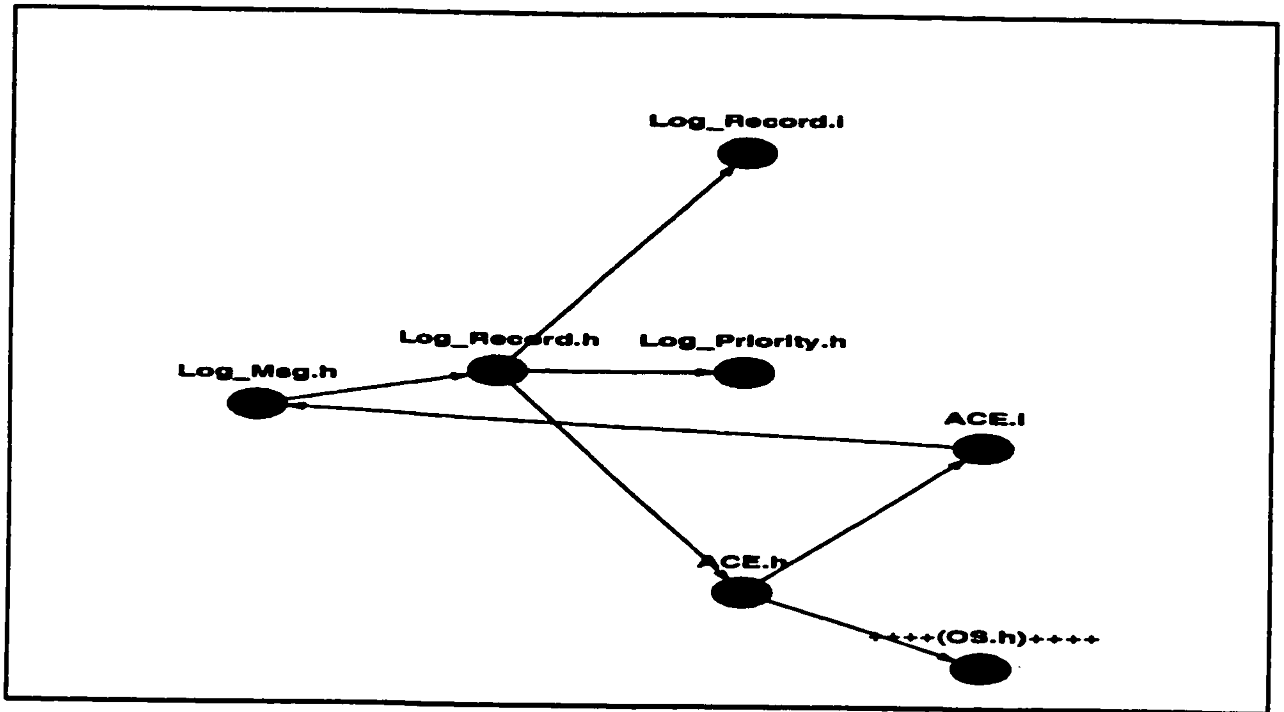


FIGURE 2–4 : Log_Msg.h File

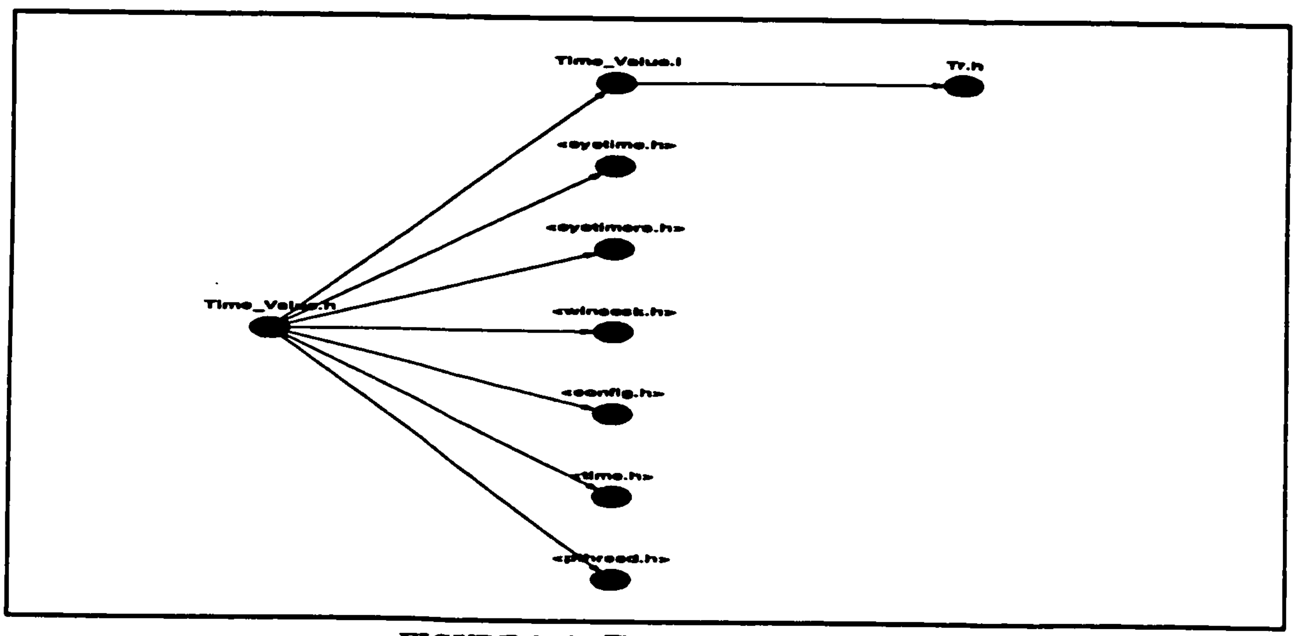


FIGURE 2–5 : Time_Value.h File

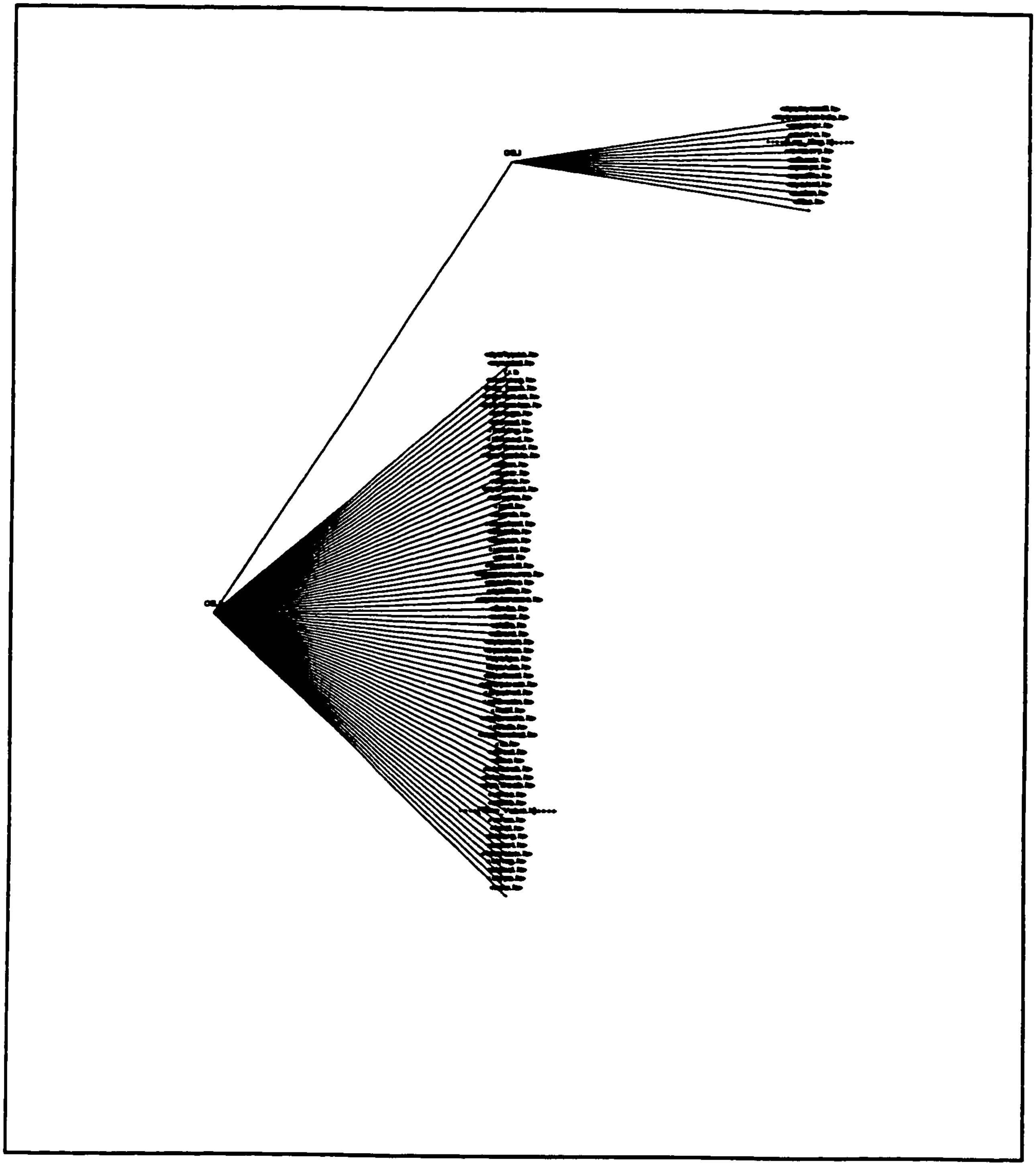

**FIGURE 2–6 :** OS.h File

# 2.4 Implementation

## 2.4.1 Inheritance

### 2.4.1.1 What Is Inheritance?

Inheritance is a relationship among components (classes) in the same system. In object–oriented technology, a class is the basic construct that is used to build programs. A class is called a parent class (also called super class, base class, ancestor class) of one or more other children classes (also called sub–classes, derived, inheriting classes) if it is inherited by these children classes [19, 26]. A parent class has two possible types of clients: sub–classes and objects (i.e. instantiations) [55]. A class has three possible sections: public, protected, and private. General speaking, inheritance is intended to facilitate programming by extension rather than programming by re–invention [6, 13, 15, 17, 18]. In chapter 5, we will show how this relation should be used among components.

Some programming languages such as Java [11–11, 16], support only single inheritance type. While other languages such as C++, support both single and multiple inheritance types. In the case of single inheritance, a class inherits from only one class (i.e. the inheriting class has only a single parent). While in the case of multiple inheritance, a class inherits from two or more other classes (i.e. the inheriting class has multi–parents) [6, 13]. Inheritance can be safe if it reveals only the behavior of the inherited classes to the inheriting ones. In other words, inheritance can be safe if the basic software design concept of information hiding is kept maintained . For further detail on inheritance, the following references are recommended [5, 13–16, 18–19, 22, 30–31, 39–40, 44, 64–66].

### 2.4.1.2 Does Multiple Inheritance Cause Any Additional Difficulties?

With single inheritance, any name can be simply resolved to a class member by simply accessing the first name encountered for data members and by accessing the first signature match encountered for methods. Since several parents can declare a member within a multiple inheritance hierarchy, it becomes hard to decide which one to choose from these declarations. As a result, multiple inheritance causes additional difficulties to object–oriented programming [24].

#### 2.4.1.3 Why Do We Need To Study Inheritance In CO–Toolkits?

Inheritance is used in almost every Object–Oriented–program. It is intended to enable developers to reuse existing components instead of completely writing their programs from the scratch. However, inheritance is not always safe. Regardless of the selected programming language, every time inheritance is used, developers should be concerned about the degree of safety that their programs will maintain [55, 56]. In this research, we want to study inheritance in order to show how to formalize rules for safe usage of inheritance among CO–Toolkits' components.

#### 2.4.1.4 ACE Inheritance

We have used Together/C++ [34] to extract and visualize inheritance relation that exist among some of ACE components. Given that a CO–Toolkit is composed of a set of frameworks that are intended to be independent from each other, we decided to look at inheritance relations that exist in two ACE frameworks: the REACTOR, and the MEMORY (*c.f. Figures 2–7, and 2–8*).

Figure 2–7 shows the inheritance relation that exists among the REACTOR framework's components. The component called ACE_Event_Handler is the parent of five other components (i.e. ACE_Proactor, ACE_Sig_Adapter, ACE_ReactorEx_Notify, ACE_Reactor_Notify, and ACE_Event_Handler_T). Consequently, the component called ACE_Sig_Handler is the parent of the ACE_Sig_Handlers component. It is important to mention at this point that the REACTOR framework contains many other components that are not shown in Figure 2–7 simply because they are not involved in inheritance.

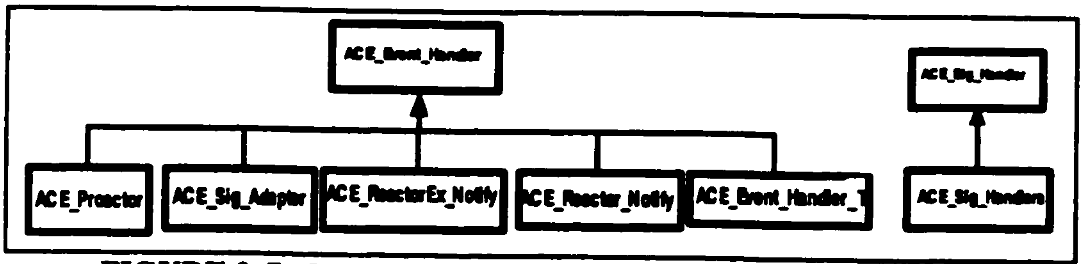


**FIGURE 2–7 :** Inheritance in the REACTOR framework

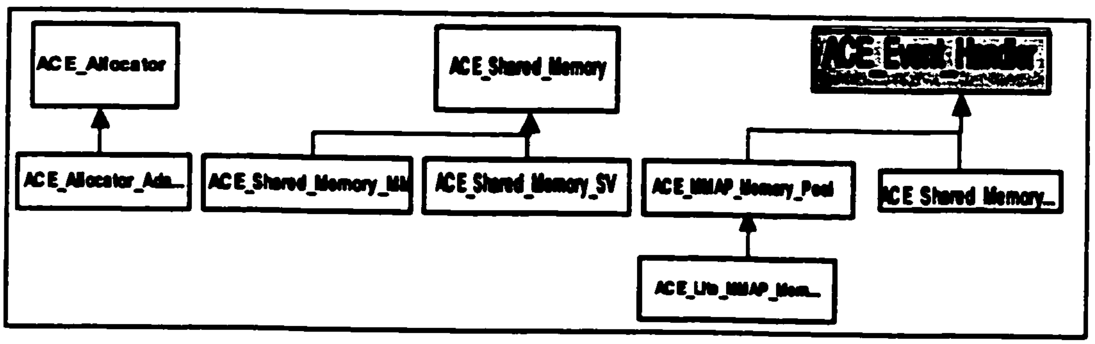


**FIGURE 2–8 :** Inheritance in the MEMORY framework

Figure 2–8 shows the inheritance relation among the MEMORY framework's components. In this Figure, the ACE_Event_Handler component (i.e. the shaded one) belongs to a different framework (i.e. the REACTOR framework).

## 2.4.2 Association

### 2.4.2.1 What Is Association?

Association is a relation between two components in the same system. We consider a pair of components to be associated with each other if at least one of them contains (or has a reference to) an object that is instantiated from the other component [6, 29]. Association can be uni or bi–directional.

#### 2.4.2.2 Why Do We Need To Study Association In CO–Toolkits?

Association is a relation that is common in almost every program, therefore it is important to show how to formalize rules for using association such that CO–Toolkits structures remain simple.

#### 2.4.2.3 ACE Association

We have used Together/C++ [34] to extract and visualize the association relations that exist in the REACTOR and MEMORY frameworks of ACE. These relations are illustrated in Figures 2–9 and 2–10 respectively.

In Figure 2–9, we can notice the following: (a) Several classes associate with themselves. (b) A class could associate with another class or with a struct (i.e. a C++ construct). (c) Only uni–directional association appears. (d) In one case, a class associates with a stranger class that does not belong to the same framework. (e) A class may associate with several other classes (i.e. multiple association). Similar observations can be seen in Figure 2–10 as well.

Association can be a uni–directional or bi–directional relation. For example, when component A contains an object from component B, but B does not have any objects of A, then we can say that component A has a uni–directional association relation with component B. If both components contain object(s) from each other, then the two components have a bi–directional association relation with each other.

In order to keep the frameworks independent from each other, association (i.e. similar to inheritance) should not be permitted among components from different frameworks. Within the same framework, association among components can be either uni–directional or bi–directional. However, association between components from frameworks and classes from shared pools should be uni–directional since that we do not want a shared class to depend on any component.

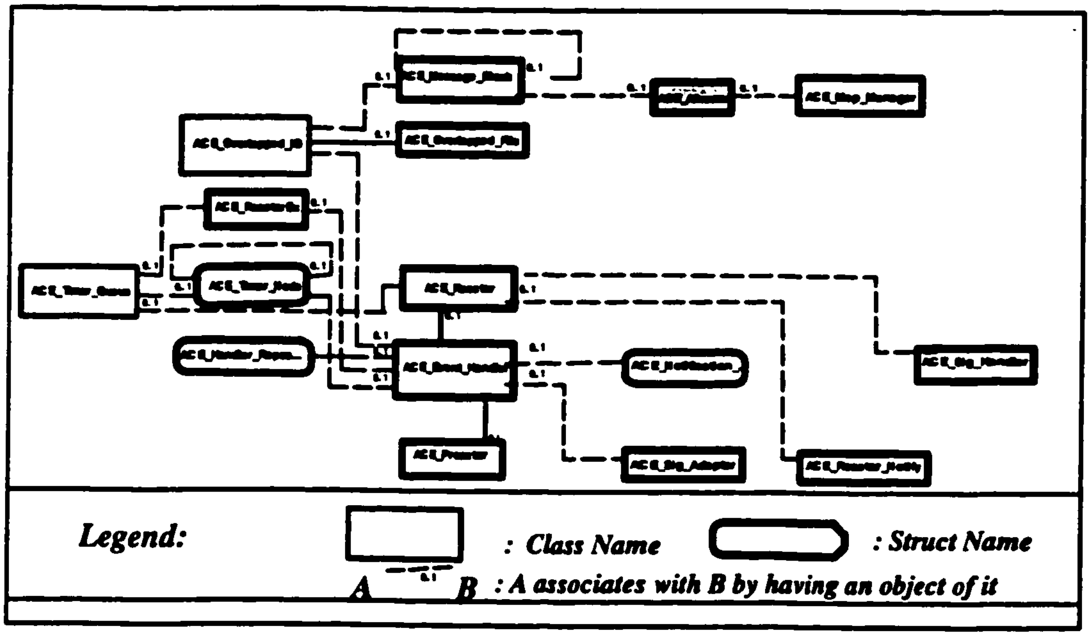


**FIGURE 2–9 :** Association in the REACTOR framework

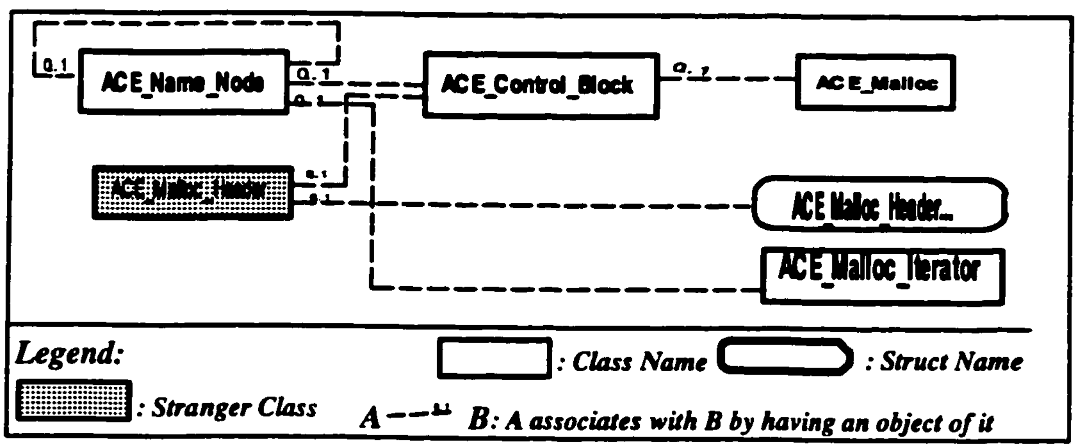


**FIGURE 2–10 :** Association in the MEMORY framework

## 2.5 Documentation

### 2.5.1 What Documents Did We Find With ACE?

Besides the code of ACE, we found a set of papers. One of them [43] provides an overview of ACE, while each of the others describes a part ACE [44–52]. When we started this research, we found it much easier to look at ACE through its code than through its papers. We think that ACE papers were written for professionals who are already familiar with ACE. Also, we found a set of manuals that basically tell what file to "#include" for a given component.

### 2.5.2 Are Such Documents Enough From The User Point Of View?

We believe that the answer for such question is NO. Our experience and the experience of others [9] show that toolkits like ACE are not something easy to use. In most cases, this would encourage developers to avoid the difficulties of using CO–Toolkits by simply not using them.

ACE was presented by people from the industry as a good tool that they could use. However, it became clear (i.e. only by use) that it was not good enough for widespread use.

## 2.6 Conclusions

In this chapter, we have:

(1) Described ACE, the case study that is considered for this research.

(2) Discussed the need for:

*i–* Studying CO–Toolkits' file structures (*c.f. section 2.3.1*).

*ii–* Studying the inheritance relation among Components (*c.f. section 2.4.1.3*).

*iii–* Studying the association relation among Components (*c.f. section 2.4.2.2*).

(3) Presented the:

*i–* File structure that is used in ACE (*c.f. section 2.3.2*).

*ii–* Inheritance relations that are used in some of ACE frameworks (*c.f. section 2.4.1.4*).

*iii–* Association relations that are used in some of ACE frameworks (*c.f. section 2.4.2.3*).

*iv–* ACE documents (c.f. section 2.5)

# CHAPTER 3

# ASSUMPTIONS FOR GOOD DESIGN

## 3.1 Overview

Having looked at ACE as an example to start this research with, the next step has been to review the software engineering principles and derive a set of realistic assumptions for good design practice.

This chapter describes the main software engineering principles (i.e.Ghezzi et al text book [21] , Parnas et al papers [36, 37, 38]) that we used to derive our assumptions. These assumptions are described in this chapter as well (*c.f. section 3.3*). Throughout this thesis, we used these assumptions to derive and formalize rules for : (i) organizing CO–Toolkits' file structures (*c.f. chapter 4*), (ii) implementing CO–Toolkits' components (*c.f. chapter 5*), and (iii) documenting CO–Toolkits (*c.f. chapter 6*).

This chapter has four sections. The next section describes the main software design principles that we used in this research to make the assumptions for good design practice. These assumptions are described in the third section. The fourth section concludes what has been done in this chapter.

## 3.2 Software Engineering Principles

This section describes the main software design principles that have been considered when making the initial assumptions for good design (*c.f. section 3.3*). Separation of concerns (*c.f. section 3.2.1*) can be seen as the primary software design principle. It can be achieved by employing principles such as modularity (*c.f. section 3.2.2*) and abstraction (*c.f. section 3.2.1.3*) with information hiding (*c.f. section 3.2.1.2*) being the guiding principle when identifying modules and designing interfaces. The primary goal behind having such design principles is to master the software complexity by separating concerns.

As a result of mastering software complexity, it becomes easier to make changes because each concern can be changed by changing only one part of the system (*c.f. section 3.2.4*). This gives another view of information hiding and abstraction. We hide what is likely to change, and the abstraction contains what we do not expect to change.

### 3.2.1 Separation of Concerns

#### 3.2.1.1 Overview

This principle can be seen as the primary principle for software design . By separating concerns, we can concentrate on each concern without concerning about the details of other perspectives [21]. For example, a software system can be evaluated from separate perspectives such as: (i) how it has been documented, (ii) how its file structure has been organized, (iii) how it has been implemented, (iv) what functions it offers, (v) how reliable it is, (vi) how efficient it is, (vii) how correct it is, (viii) how friendly its user interface is, etc.. In this research, CO–Toolkits have been viewed from three separate perspectives (i.e. file structures, implementation structures, and documentation as in chapters 4, 5, and 6 respectively).

#### 3.2.1.2 Information Hiding

Information hiding is another view of the "separation of concerns" principle. Concerns are separated by hiding design details of each software module from the designers of other modules. Another view of information hiding is that we hide what is likely to change. Also, "information hiding" can be seen as a means to separate the user interface of a given module from its implementation details. For more details on "information hiding", readers can refer to Parnas et al works [36, 37]. In this research, we have enforced the data structure implementation of any component to be provided in the internal file of that component (i.e. as in chapter 4's view for CO–Toolkits' file structures). Also, we have enforced the public section (i.e. as per the C++ terminology) of any internal class (i.e. of any component) to contain only the interface descriptions (i.e. as in chapter 5's view for CO–Toolkits' design and implementation).

#### 3.2.1.3 Abstraction

Abstraction is also, another view of "separation of concerns". Abstraction is the way to communicate what can not and should not be hidden. Modules must communicate, therefore, we abstract from what is hidden and what remains is called "the abstraction". Another view of abstraction is that the abstraction of a system contains what is not expected to change.

### 3.2.2 Modularity

The general purpose of this principle is to reduce what we have to think about at any one time. Modularity can help us to compose systems from smaller modules. Also, it can help us to decompose systems into smaller modules (parts) such that it becomes easier to understand each module by itself [36]. The "main benefit of this principle is that it allows the principle of separation of concerns to be applied in two phases: when dealing with the details of each module in isolation (and ignoring details of other modules) and when dealing with the overall characteristics of all modules and their relationships in order to integrate them into a coherent system" [21]. In this research, we have viewed a CO–Toolkit as a set of independent frameworks such that each of them is composed of a set of collaborating components.

### 3.2.3 Formality

This principle requires the software process to be driven and evaluated by mathematical laws [21]. Providing formalized and correct specification for software design and interface (i.e. as in Parnas et al [39, 40]) can help developers to produce systems that can function as expected. Formality can be seen as a means to achieve other ends such as precision, compatibility, ease of use, etc.. In this research, we have formalized the rules and guidelines for constructing CO–Toolkits' file structures (*c.f. chapter 4*), implementing CO–Toolkits (*c.f. chapter 5*), and documenting Co–Toolkits (*c.f. chapter 6*).

### 3.2.4 Mastering Complexity And Design For Change

Mastering complexity in software systems is the primary goal of applying the "separation of concerns" principle. By separating concerns, it becomes easier to handle each concern by itself without concerning about the details of other concerns. As a result, it becomes easier to change any concern by only changing one part of the system without affecting the system's other parts [21, 38]. In this research, we have provided formalized rules (*c.f. chapters 4, 5, and 6*) based on which, we have built our prototypes (*c.f. appen-*

*dixes A, and B*). Each of this research's rules can be seen as a separate concern. If a rule needs to be changed, only the part that represents that rule will need to be changed.

## 3.3 Assumptions For Good Software Design Practice

This section describes the set of assumptions that we made based on the software engineering principles described in the previous section as well as based on our experience with ACE as explained in the previous chapter.

We think that these assumptions are good for design practice. However, we do not claim that they are firm ones that should be used "as is" by developers.

In [28], Lakos provides principles and rules for how to construct components. These principles discuss issues that are mainly arbitrary and C++ language–dependent. As our assumptions and rules are intended to be general, and as this research is intended to show how to derive and formalize rules, not to derive language–dependent ones, we think that discussing Lakos rules is out of the scope of this research. However, showing how to formalize Lakos rules (i.e. as indicated in the "Future Work" section of chapter 8) can be a good research opportunity in the future.

### 3.3.1 General Assumptions

This section describes the set of initial assumptions that were considered in the three views (*c.f. chapters 4, 5, and 6*).

As it is important to consider the software engineering principles in the development process, we think that the following concerns need to be considered when organizing, designing, and documenting CO–Toolkits: (i) the file structures used for the implementation of CO–Toolkits, (ii) the clustering of CO–Toolkits' components into frameworks, (iii) the dependencies among frameworks inside the same CO–Toolkit, and (iv) the internal design of components and the dependencies among them.

As we are studying software systems that are complicated and hard to understand because of the many existing relations among their components, we need to use existing tools in order to abstract and visualize these relations.

As per the Formality principle (*c.f. section 3.2.3*), having formalized and precise rules in–place before developing checking tools can help in speeding up the development process. Checking tools can be used then to check systems and report problems back to developers.

A CO–Toolkit can be composed of a set of frameworks and a set of shared pools such that: (i) each framework can be composed of a set of components, and (ii) each shared pool can be composed of a set of classes and other entities that are shared by a number of the frameworks. A framework can share (i.e. have access to) more than one shared pool. This has been based on: (i) the modularity principle (*c.f. section 3.2.2*) in the sense that a CO–Toolkit has a modular structure, (ii) the separation of concerns principle (*c.f. section 3.2.1*) in the sense that the shared elements are separated from the frameworks, and (iii) our experience with ACE in the sense that ACE is composed of a set of frameworks.

## 3.3.2 Assumptions For Good File Structures

This section describes the assumptions that are considered for the CO–Toolkits' file structures view (*c.f. chapter 4*).

A CO–Toolkit's file structure can be simplified by grouping the files that are shared among the CO–Toolkit's frameworks into a set of shared pools (*c.f. section 3.2.1:*

*separation of concerns*). This is to minimize duplication of files and to eliminate dependencies among frameworks such that it becomes possible to study, analyze, and use each of these frameworks by itself (*c.f. section 3.2.4: mastering complexity and design for change*).

A number of files can be used to implement a component. As indicated in the overview section of chapter 1, we use three files for each component; top, intermediate, and internal. The top file is to hold the component's main part, the intermediate file is to hold the component's interface, and the internal file is to hold the implementation of that component. This has been based on our experience with ACE where up to three files are used to implement each component.

In order to keep frameworks independent from each other, files can not exist at the same time in more than one framework. Shared–pools are proposed in this thesis to hold any files that need to be shared by frameworks. This has been based on: (i) the separation of concerns and modularity principles (*c.f. sections 3.2.1, and 3.2.2 respectively*) in the sense that we need to be able to decompose a CO–Toolkit into smaller modules (i.e. frameworks in this case) that can be studied separately, (ii) the objective that we need to master the complexity of our systems (*c.f. section 3.2.4*) such that it becomes possible to make changes in one framework without affecting the others.

As the "designing software for change and mastering its complexity" is the key force in this research, we think that it is not a good practice to permit either conditional (i.e. a header file is to be ignored if it has been already included in the same inclusion sequence) or unconditional (i.e. as opposed to conditional) loops in file inclusion. Such inclusion loops can make it hard to track down the code parts that need to be changed.

Concerning the use of top files, we assumed that it is a good practice to satisfy the following:

(1) A top file should not be included by any other file. This is to keep file structures simple and easy to understand,

(2) A top file can include only its intermediate file. This is to help maintain design principles such as information hiding (*c.f. section 3.2.1.2*), and

(3) Only top files can "include" shared files from shared–pools. This is to minimize the external effect on components.

Concerning the use of intermediate files, we assumed that it is a good practice to enable an intermediate file to include intermediate files from other components within the same framework. This is to enable components collaboration as learned from ACE.

Concerning the use of internal files, we assumed that it is a good practice to satisfy the following:

(1) An internal file can not include any other file. This is to keep the component's data structure in one file (*c.f. section 3.2.1.2: information hiding*), and

(2) An internal file can not be included by any other file except its intermediate file. This is to ensure that a component's data structure can be accessed by its functions only.

Concerning the use of files from the shared pools, we assumed that it is a good practice to satisfy the following:

(1) enable inclusion among file within the same shared pool. This is to keep shared pools independent from each other *(c.f. section 3.2.1: separation of concerns, and section 3.2.2: modularity)*,

(2) have each shared–pool be accessible by a sub–set of the frameworks. This is to maintain organized and simple file structures, and

(3) have any library file that is provided by the platform system to set in the main shared–pool that is shared by all frameworks. This is to enable the usability of system libraries by all frameworks.

### 3.3.3 Assumptions For Good Implementation Structures

This section describes the initial assumptions that are considered for the view on CO–Toolkits' design and implementation (*c.f. chapter 5*).

As defined in the overview section of chapter 1, a component can be implemented using a main part and a set of internal classes. This is to make it possible to either use each component as a stand–alone application, or invoke its internal classes by other components.

We assumed that the implementation of CO–Toolkits needs to address the following: (i) their frameworking scheme, (ii) the internal design of their components, and (iii) the dependencies (i.e. relations) among their components. This has been based mainly on the: (i) abstraction principle (*c.f. section 3.2.1.3*) in the sense that a CO–Toolkit can be seen (i.e. abstracted) as a set of frameworks and also a framework can be seen as a set of components, (ii) modularity principle (*c.f. section 3.2.2*) in the sense that a CO–Toolkit has a modular framework structure, and (iii) objective of mastering complexity and design for change (*c.f. section 3.2.4*) in the sense that we want to make it easy to understand and use CO–Toolkits.

Concerning the use of the inheritance relation among components, we assumed that it is a good practice to satisfy the following:

(1) Components from different frameworks can not inherit from each other. This is to keep frameworks independent from each other (*c.f. section: 3.2.2 modularity, and section: 3.2.4 mastering complexity and design for change*),

(2) A component can inherit from only one other component from the same framework. This is to maintain simple inheritance structures (*c.f. section: 3.2.4 mastering complexity and design for change*),

(3) Only public inheritance can exist among components. This is a limitation that we propose on the use of inheritance among components. This limitation helps in maintaining a safe use of inheritance. Protected and private inheritance can reveal internal structures of components and that violates the information hiding principle (*c.f. section 3.2.1.2*).

(4) A component's internal class can inherit from only one other internal class from the same component. This is another limitation that we propose on the use of inheritance in order to maintain simple inheritance structures (*c.f. section: 3.2.4 mastering complexity and design for change*).

(5) Inheritance can not exist among shared classes from different shared pools. This is to keep shared–pools independent from each other. This is mainly based on the separation of concerns, modularity, and design for change principles.

(6) A shared class can inherit from only one other shared class from the same shared pool. This is to maintain only single inheritance relations among shared classes (*c.f. section: 3.2.4 mastering complexity and design for change*).

Concerning the use of the association relation among components, we assumed that it is a good practice to satisfy the following:

(1) Association can not exist among components from different frameworks. This is to keep frameworks independent from each other. As with inheritance, this assumption is mainly based on the modularity, separation of concerns, and design for change principles.

(2) Association and inheritance relations can not exist at the same time among a component's two internal classes. This is for convenience in order to maintain simple inheritance and association structures.

### 3.3.4 Assumptions For Good Documentation

This section describes the assumptions that are considered for the view on CO–Toolkits' Documentation (*c.f. chapter 6*). Since developers of CO–Toolkits often do not provide proper documentation for their systems, it becomes hard to understand and use these systems. Reusability of components is intended to simplify and enhance the process of software development. However, it often takes developers a long time to understand the CO–Toolkits they want to use. As a result of not having proper and formal documentation, we think that using existing components from CO–Toolkits often slows down the process of development rather than speeding it up.

We assumed that the "table based Documentation methods" [7, 36–40] can be used to document CO–Toolkits. This is based on the fact that these methods have been in use during the last 23 years and different people from different countries have used them. Examples are the A7 project [7], and the work by Parnas et al [36–40]. Such documentation methods require a lot of time and effort to be invested up–front. However, if automation tools such as the Table Tool System (i.e. TTS for short) [1] are used to produce such formal document, the up–front investment can be substantially reduced. Also, using such documentation methods can reduce the cost of developing and maintaining systems code. Our experience with building the prototypes for this research has shown that having formal specifications up–front can make it easy to develop the system's code. Investigating the effectiveness of these documentation methods is out of the scope of this research.

## 3.4 Conclusions

In this chapter, we have:

(1) Discussed the main software design principles (*c.f. section 3.2*) that we considered when we made the initial assumptions for this research.

(2) Made the following assumptions for CO–Toolkits' design practice:

*i–* General assumptions (*c.f. section 3.3.1*) that were considered in the three views discussed in chapters 4 through 6.

*ii–* Assumptions for good file structures (*c.f. section 3.3.2*) that were considered in the first view for CO–Toolkits' file structures (*c.f. in chapter 4*).

*iii–* Assumptions for good design and implementation structures (*c.f. section 3.3.3*) that were considered in the second view for CO–Toolkits' design and implementation (*c.f. chapter 5*).

*iv–* Assumptions for good CO–Toolkits' documentation practice (*c.f. section 3.3.4*) that were considered in the third view for CO–Toolkits' documentation (*c.f. chapter 6*).

# CHAPTER 4

# FIRST VIEW: FORMALIZING RULES FOR ORGANIZING FILE STRUCTURES OF CO–TOOLKITS

## 4.1 Overview

The next section describes in detail our view for CO–Toolkits' file structures. This view is based on the assumptions for good design practice that we made in chapter 3 (*c.f. sections 3.3 and 3.3.2*). Section 2.3.1 of chapter 2 explains why we need to study CO–Toolkits' file structures. Appendix A describes a prototype system that we built based on this view. Two artificial examples and a third one on a sample ACE file structure were used to demonstrate this prototype. Section 8.2 of chapter 8 discusses the limitations of our views (*c.f. chapters 4 through 6*) including this file structures one.

## 4.2 Towards A Mathematical View For CO–Toolkits' File Structures

### 4.2.1 Purpose

This view is intended to formalize a set of rules and guidelines for how to maintain well–organized file structures in CO–Toolkits. These rules and guidelines are based on the initial assumptions for good design practice that we made in chapter 3 (*c.f. sections 3.3 and 3.3.2*). This view can be expanded by adding new rules that we are not aware of at this time.

This view can be used to develop a tool that can be used to check CO–Toolkits' file structures and report problems back to developers. To support this claim, we have built a simple prototype system (*c.f. Appendix A*) based on this view and demonstrated it by using two artificial examples and a third example on a sample ACE file structure.

### 4.2.2 The View Structure

The CO–Toolkit's file structure that we considered for this research (*c.f. Figure 4–1*) is simple. A CO–Toolkit is represented by two disjoint groups of files, one group is used to implement the frameworks, and the second is used to implement the shared–pools. The first group is also divided into several subsets of files such that each of them is used to implement a particular framework. Also, the second group is divided into several subsets of files such that each of them is used to implement a particular shared–pool. Consequently, each framework's set of files is divided into a number of subsets such that each of them is used to implement a particular component. Also, each component's set of files contains three files that share the same filename, but with different extensions (.top refers to the component's top file, .inm refers to the component's intermediate file, and .int refers to the component's internal file). Readers can refer to the lexicon of this view (*c.f. section 4.2.4*) for the meanings of the variables and symbols that appear in the various formulae throughout this section.

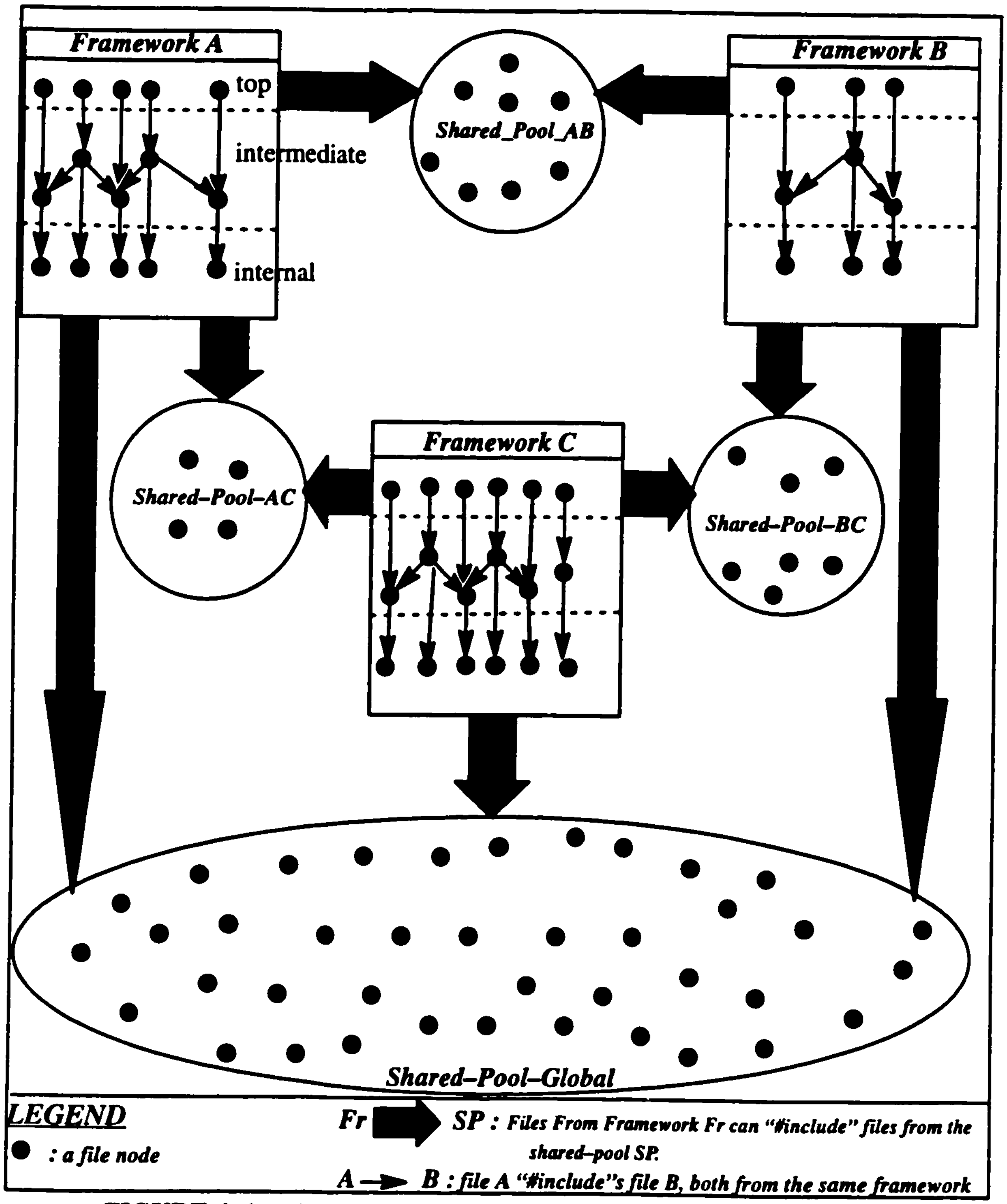


**FIGURE 4–1 :** Motivating example: Model for CO–Toolkits' file structures

In formula (4–1), the variable *CO_Files* is used to represent the set of all files used to implement the given CO–Toolkit. This *CO_Files* is represented by the union of the following two subsets of files:

(1) $\bigcup_{i=1}^{F} Fr\_Files_i$: this element represents the union of the sets of files that are used to implement the frameworks. $Fr\_Files_i$ is used to represent the set of files that are used to implement the $i^{th}$ framework given that $1 \leq i \leq F$, where $F$ represents the number of frameworks.

(2) $\bigcup_{k=0}^{S} Sh\_Files_k$: this element represents the union of the sets of files that are used to implement the shared–pools. $Sh\_Files_k$ is used to represent the set of files that are used to implement the $k^{th}$ shared–pool given that $0 \leq k \leq S$, where $S$ represents the number of shared–pools. We should notice that a given CO–Toolkit may have no shared–pools at all.

$$CO\_Files \equiv \bigcup_{i=1}^{F} Fr\_Files_i \cup \bigcup_{k=0}^{S} Sh\_Files_k \qquad (4\text{–}1)$$

Given that each of the $F$ frameworks is composed of a set of components, $C\_Files_{ij}$ in (4–2) is used to represent the subset of files that are used to implement the $j^{th}$ component of the $i^{th}$ framework such that $1 \leq j \leq C_i$, where $C_i$ represents the number of components in the $i^{th}$ framework – throughout this thesis, we will call it for short, the $ij^{th}$ component. Readers should notice that a framework should contain at least one component.

$$Fr\_Files_i \equiv \bigcup_{j=1}^{C_i} C\_Files_{ij} \qquad (4\text{–}2)$$

Given that there are $S$ shared–pools, $Sh\_File_{kl}$ in (4–3) is used to represent the $l^{th}$ file in the $k^{th}$ shared–pool such that $0 \le l \le s_k$, where $s_k$ represents the number of files in the $k^{th}$ shared–pool.

$$Sh\_Files_k \equiv \bigcup_{l=1}^{s_k} Sh\_File_{kl} \qquad (4\text{–}3)$$

In formula (4–4), each component is represented by its three files referred to by $topFile_{ij}$, $intermediateFile_{ij}$ and $internalFile_{ij}$.

(1) $topFile_{ij}$: this element represents the top (i.e. .cpp) file of the $ij^{th}$ component.

(2) $intermediateFile_{ij}$: this element represents the intermediate (i.e. .h) file of the $ij^{th}$ component.

(3) $internalFile_{ij}$: this element represents the internal (i.e. .i) file of the $ij^{th}$ component.

$$C\_Files_{ij} \equiv \{topFile_{ij}, intermediateFile_{ij}, internalFile_{ij}\} \qquad (4\text{–}4)$$

### 4.2.3 The View Characteristics

This sub–section describes the set of characteristics (i.e. rules) that we think should be maintained by the above view. These characteristics are based on the initial assumptions for good design that we made in chapter 3 (*c.f. sections 3.3 and 3.3.2*). Readers can refer to the lexicon of this view (*c.f. 4.2.4*) for the meanings of the variables and symbols that appear in the various formulas throughout this section.

In formula (4–5), the predicate *GoodFileStructure*() reflects what we mean by having well–organized file structure. This predicate is formed by taking the conjunction of the three sub–predicates that are listed next and defined in more detail in section 4.2.3.1.

(1) *OnlyFourFileCategories()*: this means that any file should be either a top, an intermediate, an internal, or a shared file.

(2) *GoodDistinctionOfFiles()*: this means that frameworks and their components should have distinctive files.

(3) *GoodFileInclusionRelation()*: this means that the overall file "#include" relation should be well–organized.

$$GoodFileStructure() \equiv \begin{bmatrix} OnlyFourFileCategories() \;\wedge \\ GoodDistinctionOfFiles() \;\wedge \\ GoodFileInclusionRelation() \end{bmatrix} \tag{4–5}$$

### 4.2.3.1 Explanation of (4–5): *GoodFileStructure()*

In formula (4–6), the definition for *OnlyFourKindsOfFiles()*, the first sub–predicate of (4–5) above is final. *Category(A)* represents an auxiliary function that returns the category of file *A*, and that is defined in the lexicon of this view (*c.f. section 4.2.4*).

$$OnlyFourFileCategories() \equiv \left[\forall x, (x \in CO\_Files) \Rightarrow \left[Category(x) \in \left\{\begin{matrix} 'top', \\ 'intermediate', \\ 'internal', \\ 'shared' \end{matrix}\right\}\right]\right] \tag{4–6}$$

In formula (4–7), the definition for *GoodDistinctionOfFiles()*, the second sub–predicate of (4–5) above is not final. It is formed by taking the conjunction of the two sub–predicates that are listed next and defined in more detail in section 4.2.3.2.

(1) *Fr_SetsOfFilesShouldBeDistinct()*: this means that frameworks should have distinctive files.

(2) *ComponentsMainFilesShouldBeDistinct()*: this means that each component within a given framework should have its own top, intermediate, and internal files. These two sub–predicates are explained in this chapter.

$$GoodDistinctionOfFiles() \equiv \begin{pmatrix} Fr\_SetsOfFilesShouldBeDistinct() \wedge \\ ComponentsMainFilesShouldBeDistinct() \end{pmatrix} \tag{4–7}$$

In formula (4–8), the definition for *GoodFileInclusionRelation*(), the third sub-predicate of (4–5) above is not final too. It is formed by taking the conjunction of the two sub-predicates that are listed next and defined in more detail in section 4.2.3.3.

(1) *GoodUseOfFiles*(): this states how to make good use of each of the four possible kinds of files (top, intermediate, internal, and shared files).

(2) *InclusionLoopsMayNotExist*(): this states that inclusion loops are not permitted in CO-Toolkits file structures.

$$GoodFileInclusionRelation() \equiv \begin{pmatrix} GoodUseOfFiles() \wedge \\ InclusionLoopsMayNotExist() \end{pmatrix} \tag{4–8}$$

#### 4.2.3.2 Explanation of (4–7): *GoodDistinctionOfFiles*()

In formula (4–9), the definition for *FR_SetsOfFilesShouldBeDistinct*(), the first sub-predicate of (4–7) above is final. It means that the intersection of frameworks sets of files is an empty set (i.e. frameworks have distinctive sets of files).

$$Fr\_SetsOfFilesShouldBeDistinct() \equiv$$
$$\left[ \forall i,j, \begin{bmatrix} (1 \le i \le F) \wedge \\ (1 \le j \le F) \wedge \\ (j \ne i)) \end{bmatrix} \Rightarrow \left( Fr\_Files_i \bigcap Fr\_Files_j = \phi \right) \right] \tag{4–9}$$

In formula (4–10), the definition for *ComponentsMainFilesShouldBeDistinct*(), the second sub-predicate of (4–7) above is final too. It means that the intersection of components subsets of files is an empty set too.

$$ComponentsMainFilesShouldBeDistinct() \equiv$$
$$\left( \forall i, (1 \le i \le F) \Rightarrow \left( \forall j, (1 \le j \le C_i) \Rightarrow \left( \bigcap C\_Files_{ij} = \phi \right) \right) \right) \tag{4–10}$$

#### 4.2.3.3 Explanation of (4–8): *GoodFileInclusionRelation()*

In formula (4–11), the definition for *GoodUseOfFiles()*, the first sub–predicate of (4–8) above is not final. It is formed by taking the conjunction of the four sub–predicates that are listed next and defined in more detail in section 4.2.3.4.

(1) *GoodUsOf_top_Files()*: this states how top (i.e. .cpp) files should be used.

(2) *GoodUsOf_intermediate_Files()*: this states how intermediate (i.e. .h) files should be used.

(3) *GoodUsOf_internal_Files()*: this states how internal (i.e. .i) files should be used.

(4) *GoodUsOf_shared_Files()*: this states how shared files should be used.

$$GoodUseOfFiles() \equiv \begin{bmatrix} GoodUseOf\_top\_Files() & \wedge \\ GoodUseOf\_intermediate\_Files() & \wedge \\ GoodUseOf\_internal\_Files() & \wedge \\ GoodUseOf\_shared\_Files() & \end{bmatrix} \qquad (4\text{–}11)$$

In formula (4–12), the definition for *InclusionLoopsMayNotExist()*, the third sub–predicate of (4–8) above is final. It states that inclusion loops are not permitted in file structures. This covers the special inclusion loops case of bi–directional inclusions.

Let $Loop \equiv \exists fPA \left| \begin{bmatrix} \left(\left(\forall j, (1 \le j < n) \Rightarrow \left(Includes\left(fPA[a_j], fPA[a_{j+1}]\right)\right)\right)\right) \wedge \\ \left(Includes\left(fPA[a_n], fPA[a_1]\right)\right) \end{bmatrix}\right.$, <u>where</u>

$fPA \equiv SEQUENCE[a_1, a_2, \ldots, a_n]\ of\ FileNames$, then

$$InclusionLoopsMayNotExist() \equiv \left(\forall n, (1 \le n \le NumberOfFilesInCO\_Files) \Rightarrow \neg Loop\right) \qquad (4\text{–}12)$$

#### 4.2.3.4 Explanation of (4–11): *GoodUseOfFiles()*

In formula (4–13), the definition for *GoodUseOf_top_Files()*, the first sub–predicate of (4–11) above is not final. It is formed by taking the conjunction of the three sub–predicates that are listed next and defined in more detail in section 4.2.3.5.

(1) *NoFilesMayIncludeA_top_File*(): this states that a top file can not be "#include"d by any other file.

(2) *FromSame_Fr_A_top_FileMayIncludeOnlyIts_intermediate_File*(): this states that a top file can "#include" only the intermediate file that shares the same name (i.e. belong to the same component).

(3) *A_top_FileMayInclude_shared_FilesFromPermittedSharedPools*(): this states that a top file can "#include" shared files from accessible shared–pools.

$$GoodUseOf\_top\_Files() \equiv \left[\begin{array}{l} NoFileMayIncludeA\_top\_File() \ \wedge \\ A\_top\_FileMayIncludeOnlyIts\_intermediate\_FileFromTheSameFramework() \ \wedge \\ A\_top\_FileMayInclude\_sharedFiles\_OnlyFrom\_PermittedSharedPools() \end{array}\right] \quad (4\text{–}13)$$

In formula (4–14), the definition for *GoodUseOf_intermediate_Files*(), the second sub–predicate of (4–11) above is represented by the single predicate described in (1) next. The final mathematical representation for this predicate is provided in formula (4–20). Given that this predicate represents a rule for *GoodUseOf_intermediate_Files*() and that more practice may uncover more rules for using intermediate files, we prefer not to replace this predicate by its final representation in (4–20). In the case of uncovering more rules for using intermediate files, then we can simply hook them through formula (4–14).

(1) *An_intermediate_FileMayIncludeOnlyIts_internal_FileAndOther_intermediate_FilesFromSame_Fr*(): this states that an intermediate file can "#include" only the internal file of the same component, as well as other intermediate files from the same framework.

$$GoodUseOf\_intermediate\_Files() \equiv \left(An\_intermediate\_FileMayIncludeOnlyIts\_internal\_FileAndOther\_intermediate\_FilesFromSame\_Fr()\right) \quad (4\text{–}14)$$

In formula (4–15), the definition for *GoodUseOf_internal_Files()*, the third sub-predicate of (4–11) above is not final too. It is formed by taking the conjunction of the two sub-predicates that are listed next and defined in more detail in section 4.2.3.7.

(1) *An_internal_FileMayNotIncludeAnyFile()*: this states that an internal (i.e. .i) file can not "#include" any other file.

(2) *NoFileMayIncludeAn_internal_FileExceptIts_intermediate_File()*: this states that an internal file can not be "#include"d by any other file except the intermediate file that shares the same name (i.e. belongs to the same component).

$$GoodUseOf\_internal\_Files() \equiv \left[\begin{array}{l} An\_internal\_FileMayNotIncludeAnyFile() \wedge \\ An\_internal\_FileCanBeIncludedOnlyByIts\_intermediate\_File() \end{array}\right] \tag{4–15}$$

In formula (4–16), the definition for *GoodUseOf_shared_Files()*, the fourth sub-predicate of (4–11) above is not final too. It is represented by a single predicate. The justification for not replacing this predicate by its final mathematical representation in formula (4–21) follows from the justification for (4–14) above.

(1) *A_shared_FileMayIncludeOnlyOther_shared_FilesFromSame_SP()*: this states that a shared file can "#include" other shared files from the same shared-pool.

$$GoodUseOf\_shared\_Files() \equiv \left(A\_shared\_FileMayIncludeOnlyOther\_shared\_FilesFromSame\_SP()\right) \tag{4–16}$$

#### 4.2.3.5 Explanation of (4–13): *GoodUseOf_top_Files()*

In formula (4–17), the definition for *NoFileMayIncludeA_top_File()*, the first sub-predicate of (4–13) above is final. This states that a top file can not be "#include"d by any other file. ***Includes***$(A, B)$ is a boolean auxiliary function that returns TRUE if file $A$ "#include"s file $B$. This function is defined in the lexicon of this view.

$$NoFileMayIncludeA\_top\_File() \equiv$$
$$\left[\forall x,y, \begin{bmatrix}(x \neq y) \wedge \\ (x \in CO\_Files) \wedge \\ (y \in CO\_Files) \wedge \\ (Category(x) = 'top')\end{bmatrix} \Rightarrow (\neg Includes(x,y))\right] \qquad (4\text{–}17)$$

In formula (4–18), the definition for *FromSame_Fr_A_top_FileMayIncludeOnlyIts_intermediate_File()*, the second sub–predicate of (4–13) above is final too. This states that a top file may "#include" from the same framework only the intermediate file that shares the same filename (i.e. belongs to the same component). ***FileName(A)*** is an auxiliary function that returns the filename of file *A*. This function is defined in the lexicon of this view.

Let $P1 \equiv \begin{bmatrix}(x \in Fr\_Files_i) \wedge \\ (Category(x) = 'top') \wedge \\ (Includes(x,y))\end{bmatrix}$, and $P2 \equiv \begin{bmatrix}(y \in Fr\_Files_i) \wedge \\ (Category(y) = 'intermediate') \wedge \\ (FileName(y) = FileName(x))\end{bmatrix}$, then

$$A\_top\_FileMayIncludeOnlyIts\_intermediate\_FileFromTheSameFramework() \equiv$$
$$(\forall i, (1 \leq i \leq F) \Rightarrow (\forall x,y, P1 \Rightarrow P2)) \qquad (4\text{–}18)$$

In formula (4–19), the definition for *A_top_FileMayInclude_shared_FilesFromPermittedSharedPools()*, the third sub–predicate of (4–13) above is final too. This states that a top file may "#include" a shared file from any permitted shared–pool. ***Fr_Name(A)*** is an auxiliary function that returns the name of the framework to which file *A* belongs. This function is defined in the lexicon of this view.

Let $P1 \equiv \left[\begin{array}{l}(x \in Fr\_Files_i) \wedge (y \in Sh\_Files_k) \wedge \\ (Category(x) = 'top') \wedge (Category(y) = 'shared') \wedge \\ (Includes(x,y))\end{array}\right]$, and

$P2 \equiv (Fr\_Name(boldFr\_Files_i) \in Frs\_PermittedToAccessSP(k))$ then

$$A\_top\_FileMayInclude\_sharedFiles\_OnlyFrom\_PermittedSharedPools() \equiv \left(\forall i,k, \left(\begin{array}{l}(1 \le i \le F) \wedge \\ (0 \le k \le S)\end{array}\right) \Rightarrow (\forall x,y, P1 \Rightarrow P2)\right) \quad (4\text{–}19)$$

#### 4.2.3.6 Explanation of (4–14): *GoodUseOf_intermediate_Files()*

In formula (4–20), the definition for *An_intermediate_FileMayIncludOnlyIts_internal_FileAndOther_intermediate_FilesFromSame_Fr()*, the second sub-predicate of (4–14) above is final too. This states that an intermediate file can "#include" only the internal file that shares the same filename (i.e. of the same component) as well as other intermediate files from the same framework.

Let $P1 \equiv \left[\begin{array}{l}(x \in Fr\_Files_i) \wedge \\ (Category(x) = 'intermediate') \wedge \\ (Includes(x,y))\end{array}\right]$, $P2 \equiv \left[\begin{array}{l}\left(\begin{array}{l}(FileName(y) = FileName(x)) \wedge \\ Category(y) = 'internal'\end{array}\right) \vee \\ \left(\begin{array}{l}(FileName(y) \ne FileName(x)) \wedge \\ (Category(y) = 'intermediate')\end{array}\right)\end{array}\right]$, and

$P3 \equiv ((y \in Fr\_Files_i) \wedge P2)$, then

$$An\_intermediate\_fileMayIncludeOnlyIts\_internal\_FileAndMayIncludeOther\_intermediate\_FilesFromSame\_Fr() \equiv (\forall i, (1 \le i \le F) \Rightarrow (\forall x,y, P1 \Rightarrow P3)) \quad (4\text{–}20)$$

#### 4.2.3.7 Explanation of (4–15): *GoodUseOf_internal_Files()*

In formula (4–21), the definition for *An_internal_FileMayNotIncludeAnyFile()*, the first sub-predicate of (4–15) above is final too. This states that an internal file can not "#include" any file.

*An_internal_FileMayNotIncludeAnyFile()* $\equiv$

$$\left[\forall x,y, \begin{bmatrix}(x \in CO\_Files) \wedge \\ (y \in CO\_Files) \wedge \\ (Category(x) = \text{'internal'})\end{bmatrix} \Rightarrow (\neg Includes(x,y))\right] \tag{4–21}$$

In formula (4–22), the definition for *An_internal_FileCanBeIncludedByOnlyIts_intermediate_File()*, the second sub–predicate of (4–15) above is final too. This states that an internal file can be "#include"d only by the intermediate file that shares the same file name (i.e. of the same component).

Let $P1 \equiv \begin{bmatrix}(y \in Fr\_Files_i) \wedge \\ (Category(y) = \text{'internal'}) \wedge \\ (Includes(x,y))\end{bmatrix}$, and $P2 \equiv \begin{bmatrix}(Category(x) = \text{'intermediate'}) \wedge \\ (FileName(x) = FileName(y)) \wedge \\ (x \in Fr\_Files_i)\end{bmatrix}$, then

*An_internal_FileCanBeIncludedByOnlyIts_intermediate_File()* $\equiv$

$$(\forall i, (1 \le i \le F) \Rightarrow (\forall x,y, P1 \Rightarrow P2)) \tag{4–22}$$

#### 4.2.3.8 Explanation of (4–16): *GoodUseOf_shared_Files()*

In formula (4–23), the definition for *A_shared_FileMayIncludeOnlyOther_shared_FilesFromSame_SP()*, the second sub–predicate of (4–16) above is final too. This states that a shared file can "#include" other shared files from the same shared–pool.

*A_shared_FileMayIncludeOnlyOther_shared_FilesFromSame_SP()* $\equiv$

$$\left[\forall k, (0 \le k \le S) \Rightarrow \left(\forall x,y, \begin{pmatrix}(x \in Sh\_Files_k) \wedge \\ Includes(x,y)\end{pmatrix} \Rightarrow (y \in Sh\_Files_k)\right)\right] \tag{4–23}$$

### 4.2.4 The View Lexicon

This sub–section describes the variables, symbols, acronyms, constants, and auxiliary functions that are used in formulas of the above view structure and its characteristics.

### 4.2.4.1 Variables And Acronyms

*COToolkit*: Represents any Component-Oriented Toolkit that can provide reusable components to be used to build new applications.

*CO_Files*: Represents the overall set of files that are used to implement the CO_toolkit that is under consideration.

*Fr_Files$_i$*: Represents the set of files that are used to implement $i^{th}$ framework of the CO_toolkit that is under consideration.

*C_Files$_{ij}$*: Represents the set of files that are used to implement $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

*Sh_Files$_k$*: Represents the set of files that are used to implement $k^{th}$ shared-pool of the CO_toolkit that is under consideration.

*Sh_File$_{kl}$*: Represents the $l^{th}$ file in the $k^{th}$ shared-pool.

*topFile$_{ij}$*: Represents the .cpp file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

*intermediateFile$_{ij}$*: Represents the .h file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

*internalFile$_{ij}$*: Represents the .i file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

### 4.2.4.2 Symbols And Constants

$F$: Number of Frameworks in the given CO_toolkit.

$C_i$: Number of Components in the $i^{th}$ Framework.

$S$: Number of Shared-Pools.

$s_k$: Number of Files in the $k^{th}$ Shared-Pool.

*CO_*: Component Oriented.

*Fr_*: Framework.

*SP_*: Shared-Pool.

***PA***: Path A.

***PB***: Path B.

***NumberOfFilesInCO_Files***: Represents the total number of files used to implement the given CO–Toolkit.

### 4.2.4.3 Auxiliary Functions

***Category(X)***: An auxiliary function that returns the category of the file X. It could be 'top', 'intermediate', 'internal', or 'shared'.

***FileName(X)***: An auxiliary function that returns the filename of file X.

***Fr_Name(Fr_Files$_i$)***: An auxiliary function that returns the name of the framework that is implemented using the ***Fr_Files$_i$*** file subset.

***Frs_PermittedToAccessSP(k)***: An auxiliary function that returns the set of frameworks that are permitted to access the $k^{th}$ shared–pool.

***Includes(X, Y)***: An auxiliary Boolean function that returns TRUE or FALSE. It returns TRUE when file X is "#include"ing file Y, otherwise it returns FALSE.

***Links(X, Y)***: An auxiliary Boolean function that returns TRUE or FALSE. It returns TRUE when file X contains parts of file Y, otherwise, it returns FALSE.

***SEQUENCE***$[f_1, f_2, ..., f_n]$: is used to denote a sequence of file names as per Hoffman's notation [25].

## 4.3 Conclusions

In this chapter, we have shown the following:

(1) Rules and guidelines for CO–Toolkits' file structures can be formalized using a convenient mathematical terminology and notation (*c.f. section 4.2*).

(2) Such formalized rules for file structures can be used to build a checking tool. A prototype tool that we built based on these rules is described in Appendix A.

(3) Such checking tools can be used to check CO–Toolkits' file structures and report any rule violations back to the developers of these CO–Toolkits.

Examples are provided in Appendixes A2–A4. In particular, an example on a sample ACE file structure is presented in Appendix A4.

# CHAPTER 5

# SECOND VIEW: FORMALIZING RULES FOR DESIGNING AND IMPLEMENTING CO–TOOLKITS

## 5.1 Overview

The next section describes in details our view for CO–Toolkits' design and implementation structures. This view is based on the initial assumptions for good design practice that we made in chapter 3 (*c.f. sections 3.3 and 3.3.3*). Sections 2.4.1.3 and 2.4.2.2 in chapter 2 explain why we need to study the inheritance and association relations among components. Appendix B describes a prototype system that we built based on this view for CO–Toolkits' design and implementation structures. Two artificial examples and a third one on a sample ACE inheritance structure were used to demonstrate this prototype. Section 8.2 of chapter 8 discusses the limitations of the our views (*c.f. chapters 4 through 6*) including this view for design and implementation structures.

## 5.2 Towards A Mathematical View For CO–Toolkits' Design and Implementation Structures

For convenience, we have used throughout this view a simplified indexing scheme. For example, $\mathit{Class}_{ijl}$ is used instead of $\mathit{Class}_{[i,j,l]}$ to represent the $l^{th}$ class in the

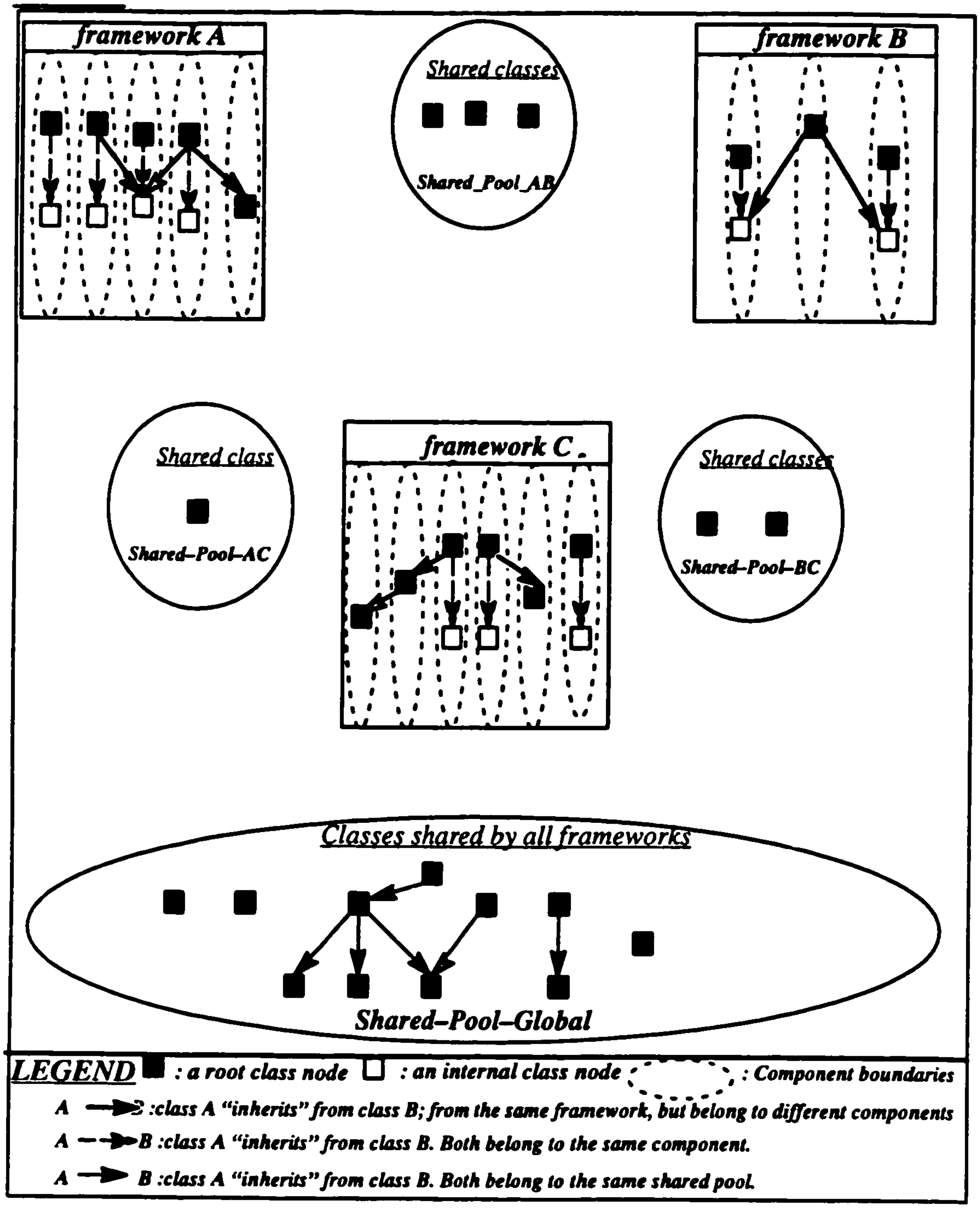


**FIGURE 5-1 :** Motivating example: Model for inheritance relation in CO-Toolkits

(1) $\bigcup_{i=1}^{F} \textit{Framework}_i$: this is used to represent the set of frameworks that are provided by the given CO–Toolkit. The $\textit{Framework}_i$ is used to represent the $i^{th}$ framework such that $1 \le i \le F$, where $F$ represents the number of frameworks in the given CO–Toolkit.

(2) $\bigcup_{k=0}^{S} \textit{SharedPool}_k$: this is used to represent the set of shared–pools that are provided in the given CO–Toolkit. The $\textit{SharedPool}_k$ is used to represent the $k^{th}$ shared–pool such that $1 \le k \le S$, where $S$ represents the number of shared–pools in the given CO–Toolkit. Readers should notice that a given CO–Toolkit should contain at least one framework.

$$\textit{COToolkit} \equiv \bigcup_{i=1}^{F} \textit{Framework}_i \bigcup \bigcup_{k=0}^{S} \textit{SharedPool}_k \tag{5–1}$$

In formula (5–1), the $\textit{Component}_{ij}$ is used to represent the $j^{th}$ component of the $i^{th}$ framework such that $1 \le j \le C_i$, where $C_i$ represents the number of components in the $i^{th}$ framework. Readers should notice that each framework should contain at least one component.

$$\textit{Framework}_i \equiv \bigcup_{j=1}^{C_i} \textit{Component}_{ij} \tag{5–2}$$

In formula (5–3), the $\textit{SharedClass}_{kj}$ is used to represent the $j^{th}$ class in the $k^{th}$ shared–pool such that $0 \le j \le c^s_k$, and the $\textit{OtherSharedEntity}_{kf}$ is used to represent the other entities that are provided in the $k^{th}$ shared pool such that $0 \le f \le e^s_k$. The symbols $c^s_k$ and $e^s_k$ respectively, represent the number of classes and the number of other shared entities that are provided in the $k^{th}$ shared–pool. Reader should notice that a shared–pool should contain at least a one shared entity.

$$SharedPool_k \equiv \begin{bmatrix} SharedClassesSet_k, \\ OtherSharedEntitiesSet_k \end{bmatrix}, \underline{\text{where}}$$

$$SharedClassesSet_k \equiv \bigcup_{j=0}^{c_k^s} SharedClass_{kj}, \text{ and } OtherSharedEntitiesSet_k \equiv \bigcup_{f=1}^{c_k^e} SharedEntity_{kf} \quad (5\text{–}3)$$

#### 5.2.2.1 Component: $Component_{ij}$ As In (5–2)

In formula (5–4), a component is specified by a tuple of its interface and implementation. The $Interface_{ij}$ and $Implementation_{ij}$ are used to respectively represent the interface and the implementation of the $ij^{th}$ component.

$$Component_{ij} \equiv \begin{bmatrix} Interface_{ij}, \\ Implementation_{ij} \end{bmatrix} \quad (5\text{–}4)$$

##### 5.2.2.1.1 Component's Interface: $Interface_{ij}$ As In (5–4)

In formula (5–5), the interface of the $ij^{th}$ component is represented by the union of its local and inherited subsets of interfaces. $LocalInterface_{ij}$ and $InheritedInterface_{ij}$ are used to represent these two subsets.

$$Interface_{ij} \equiv LocalInterface_{ij} \bigcup InheritedInterface_{ij} \quad (5\text{–}5)$$

In formula (5–6), the local interface of the $ij^{th}$ component is represented by the union of all of its local public methods' signatures. The variable $LocalPublicMethodSignature_{ijlk}$ is used to represent the signature of the $k^{th}$ public method that belongs to the $l^{th}$ class of those used to implement the $ij^{th}$ component. We should notice that $c_{ij}$ represents the number of classes that are used, along with the main section, to form the the $ij^{th}$ component, while $M_{ijl}^{p}$ represents the number of public methods that belong to the $l^{th}$ class of that component.

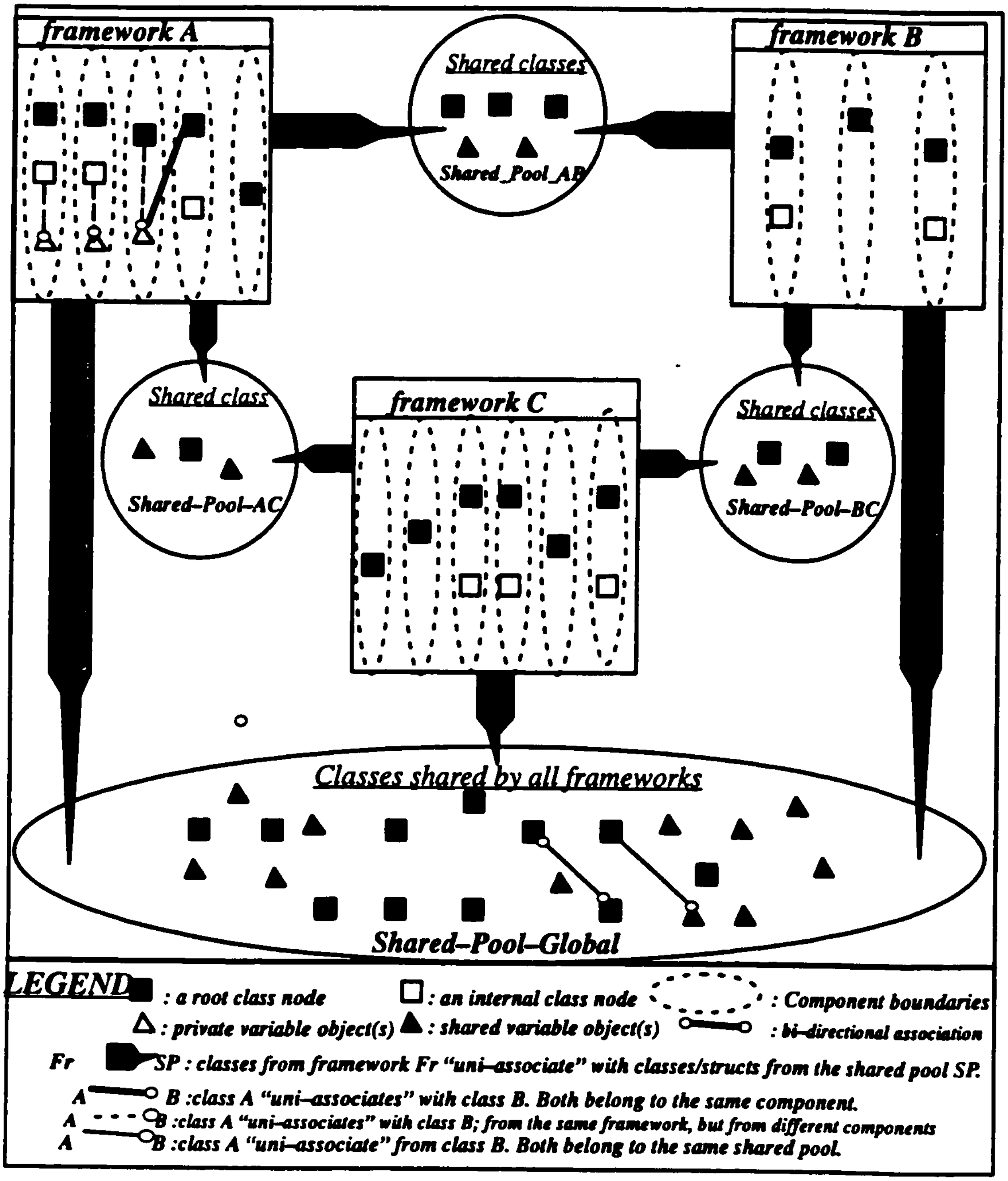


**FIGURE 5-2 :** Motivating example: Model for association relation in CO-Toolkits

$$\boldsymbol{LocalInterface}_{ij} \equiv \bigcup_{l=1}^{c_{ij}} \bigcup_{k=1}^{M^P_{ijl}} \boldsymbol{LocalPublicMethodSignature}_{ijlk} \tag{5–6}$$

In formula (5–7), the inherited interface of the $ij^{th}$ component is represented by the union of all inherited public methods' signatures. The variable $\boldsymbol{PubliclyInheritedMethodSignature}_{ijlf}$ is used to represent the signature of the $f^{th}$ publicly inherited method. This excludes those methods that are inherited among internal classes that form the same component. We should notice that $M^{P_{Inhr}}_{ijl}$ represents the number of methods that are publicly inherited by the $l^{th}$ class of the $ij^{th}$ component.

$$\boldsymbol{InheritedInterface}_{ij} \equiv \bigcup_{l=1}^{c_{ij}} \bigcup_{f=0}^{M^{P_{Inhr}}_{ijl}} \boldsymbol{PubliclyInheritedMethodSignature}_{ijlf} \tag{5–7}$$

In formula (5–8), the $\boldsymbol{LocalPublicMethodSignature}_{ijlk}$ of (5–6) is represented by the union of the data type of the value returned by the method of this signature, subset of the types of the input arguments and the subset of the types of the output arguments that are passed to and returned by that method. The auxiliary $\boldsymbol{Type}()$ function is used next to return the type of its argument.

$$\boldsymbol{LocalPublicMethodSignature}_{ijlk} \equiv$$

$$\{\boldsymbol{Type}(\boldsymbol{P\_ReturnedValue}_{ijlk})\} \bigcup \bigcup_{\alpha=0}^{pm^{IN}_{ijlk}} \boldsymbol{Type}(\boldsymbol{P\_IN}^{\alpha}_{ijlk}) \bigcup \bigcup_{\beta=0}^{pm^{OUT}_{ijlk}} \boldsymbol{Type}(\boldsymbol{P\_OUT}^{\beta}_{ijlk}) \tag{5–8}$$

As in (5–8) above, in formula (5–9), the signature of the $f^{th}$ method of those inherited by the $l^{th}$ class of the $ij^{th}$ component is also represented by the union of the data type of the returned value, the subset of the inputs' types, and the subset of the outputs' types of the $ijlk^{th}$ inherited method.

$$\boldsymbol{PubliclyInheritedMethodSignature}_{ijlf} \equiv$$

$$\{\boldsymbol{Type}(\boldsymbol{P}_{Inhr}\boldsymbol{ReturnedValue}_{ijlf})\} \bigcup \bigcup_{\alpha=0}^{pm^{IN}_{ijlk}} \boldsymbol{Type}(\boldsymbol{P}_{Inhr}\boldsymbol{IN}^{\gamma}_{ijlf}) \bigcup \bigcup_{\zeta=0}^{pm^{OUT_{Inhr}}_{ijlf}} \boldsymbol{Type}(\boldsymbol{P}_{Inhr}\boldsymbol{OUT}^{\zeta}_{ijlf}) \tag{5–9}$$

#### 5.2.2.1.2 Component's Implementation: $\boldsymbol{Implementation_{ij}}$ As In (5–4)

In formula (5–10), the implementation of the $ij^{th}$ component is represented by the implementation of its main section, and the union of the implementations of its internal classes. The $\boldsymbol{MainPart_{ij}}$ is used to represent the main part of the $ij^{th}$ component. The $\boldsymbol{InternalClass_{ijl}}$ is used to represent the $l^{th}$ class of those used to form the $ij^{th}$ component.

$$\boldsymbol{Implementation_{ij}} \equiv \begin{bmatrix} \boldsymbol{MainPart_{ij}}, \\ \boldsymbol{InternalClassesSet_{ij}} \end{bmatrix} \underline{\text{ where }} \boldsymbol{InternalClassesSet_{ij}} \equiv \bigcup_{l=1}^{c_{ij}} \boldsymbol{InternalClass_{ijl}} \qquad (5\text{–}10)$$

In formula (5–11), the $\boldsymbol{InternalClass_{ijl}}$ of (5–10) above is represented by the implementation of its public, protected, and private sections.

$$\boldsymbol{InternalClass_{ijl}} \equiv \begin{bmatrix} \boldsymbol{PublicSection_{ijl}}, \\ \boldsymbol{ProtectedSection_{ijl}}, \\ \boldsymbol{PrivateSection_{ijl}} \end{bmatrix} \qquad (5\text{–}11)$$

In formula (5–12), the $\boldsymbol{PublicSection_{ijl}}$ of (5–11) above is represented by the set of its public methods. We should notice that there are no objects/variables present in this definition. This is to exclude any data structure entities (i.e. to maintain the concept of information hiding) from the public section of any class.

$$\boldsymbol{PublicSection_{ijl}} \equiv \bigcup_{k=1}^{M^P_{ijl}} \boldsymbol{PublicMethod_{ijlk}} \qquad (5\text{–}12)$$

In formula (5–13), the $\boldsymbol{PublicMethod_{ijlk}}$ of (5–12) above is defined. Readers can refer to the lexicon of this view for more details about the variables and symbols used in this definition.

$$\boldsymbol{PublicMethod_{ijlk}} \equiv \boldsymbol{SOF_{ijlk}}\left(\boldsymbol{P\_INs_{ijlk}},\ \boldsymbol{P\_OUTs_{ijlk}}\right) \underline{\text{where}}$$

$$\boldsymbol{P\_INs_{ijlk}} \equiv \bigcup_{\alpha=0}^{pm^{IN}_{ijlk}} \boldsymbol{P\_IN^{\alpha}_{ijlk}} \text{ and } \boldsymbol{P\_OUTs_{ijlk}} \equiv \left\{\boldsymbol{P\_ReturnedValue_{ijlk}}\right\} \bigcup \bigcup_{\beta=0}^{pm^{OUT}_{ijlk}} \boldsymbol{P\_OUT^{\beta}_{ijlk}} \qquad (5\text{–}13)$$

In formula (5–14), the $ProtectedSection_{ijl}$ of (5–11) above is represented by a tuple of the set of the protected objects and the set of the protected methods of the $l^{th}$ class of the $ij^{th}$ component. We should notice that protected sections may contain objects/variables as well as protected methods.

$$ProtectedSection_{ijl} \equiv \begin{bmatrix} \bigcup_{k=0}^{O^{R}_{ijl}} ProtectedObjects_{ijlk}, \\ \bigcup_{f=0}^{M^{R}_{ijl}} ProtectedMethod_{ijlf} \end{bmatrix} \tag{5–14}$$

In formula (5–15), the $ProtectedMethod_{ijlf}$ of (5–14) above is defined. Readers can refer to the lexicon of this view for more details about the variables and symbols used in this definition.

$ProtectedMethod_{ijlf} \equiv SOF_{ijlf}(R\_INs_{ijlf},\ R\_OUTs_{ijlf})$ <u>where</u>

$$R\_INs_{ijlf} \equiv \bigcup_{alpha=0}^{rm^{IN}_{ijlf}} R\_IN^{\alpha}_{ijlf} \text{ and } R\_OUTs_{ijlf} \equiv \{R\_ReturnedValue_{ijlf}\} \bigcup \bigcup_{beta=0}^{rm^{OUT}_{ijlf}} R\_OUT^{\beta}_{ijlf} \tag{5–15}$$

In formula (5–16), the $PrivateObject_{ijlk}$ and $PrivateMethod_{ijlf}$ are used to respectively represent the private objects and methods of the $l^{th}$ class of the $ij^{th}$ component. We should notice that private sections may contain private methods besides the the private objects/variables.

$$PrivateSection_{ijl} \equiv \begin{bmatrix} \bigcup_{k=1}^{O^{V}_{ijl}} PrivateObjects_{ijlk}, \\ \bigcup_{f=0}^{M^{V}_{ijl}} PrivateMethod_{ijlf} \end{bmatrix} \tag{5–16}$$

In formula (5–17), the $PrivateMethod_{ijlf}$ of (5–16) above is defined. Readers can refer to the lexicon of this view for more details about the variables and symbols used in this definition.

$$PrivateMethod_{ijlf} \equiv SOF_{ijlf}(V\_INs_{ijlf},\ V\_OUTs_{ijlf}),\ \underline{\text{where}}$$

$$V\_INs_{ijlf} \equiv \bigcup_{\alpha=0}^{nm_{ijlf}^{IN}} V\_IN_{ijlf}^{\alpha} \text{ and } V\_OUTs_{ijlf} \equiv \{V\_ReturnedValue_{ijlf}\} \bigcup \bigcup_{\beta=0}^{nm_{ijlf}^{OUT}} V\_OUT_{ijlf}^{\beta} \qquad (5\text{–}17)$$

## 5.2.3 The View Characteristics

This sub–section describes the set of characteristics (i.e. rules) that we think should be maintained by the above view. Readers can refer to the lexicon section of this view (*cf. section 5.2.4*) for the meanings of the variables and symbols that appear in the various definitions throughout this section. As long as the predicate *GoodDesignAndImplementation()* of (5–18) is true, we say that the above view is satisfied. This predicate is represented by the next listed sub–predicate that is defined in more details in section 5.2.3.1. In order to keep this view open to any new rules that can be found in further research, we prefer not to replace this *GoodUseOfRelationsAmongComponents()* by its representation that appears in section 5.2.3.1.

(1) *GoodUseOfRelationsAmongComponents()*: as explained in (5–19), this sub–predicate is used to ensure that relations among these components are acceptable. Throughout this section, each of these sub–predicates is explained in more and more details.

$$GoodDesignAndImplementation() \equiv (GoodUseOfRelationsAmongComponenets()) \qquad (5\text{–}18)$$

### 5.2.3.1 Explanation of (5–18): *GoodDesignAndImplementation()*

In formula (5–19), the *GoodUseOfRelationsAmongComponents()* of (5–18) above, is formed by taking the conjunction of the three sub–predicates that are listed next and defined in more details in section 5.2.3.2.

(1) *SafeInheritance()*: as explained in (5–20), this sub–predicate is used ensure that inheritance is used safely from our point of view.

(2) *GoodAssociation()*: as explained in (5–21), this sub–predicate is used to ensure that association is used in a good way from our point of view.

(3) *GoodUseOfOtherRelations()*: this sub–predicate is for any potential extension to this second view of CO–Toolkits. Any relation other than inheritance and association relations that are covered by this view can be glued to this view through this sub–predicate in the future.

$$GoodUseOfRelationsAmongComponents() \equiv \begin{bmatrix} SafeInheritance() \wedge \\ GoodAssociation() \wedge \\ GoodUseOfOtherRelations() \end{bmatrix} \qquad (5\text{–}19)$$

#### 5.2.3.2 Explanation of (5–19): *GoodUseOfRelationsAmongComponents()*

In formula (5–20), the *SafeInheritance()* sub–predicate of (5–19) above, is formed by taking the conjunction of the four sub–predicates that are listed next and defined in section 5.2.3.3.

(1) *InheritanceMayNotExistAcrossFrameworks()*: as explained in (5–23), this sub–predicate is used to ensure that there is no inheritance among components that belong to different frameworks. We have recommended this since inheritance is not safe by default and in this case, inheritance would break the encapsulation at the level of frameworks.

(2) *SafeInheritance_AmongComponents_WithInTheSameFramework()*: as explained in (5–24), this sub–predicate is used to ensure that inheritance is safe whenever used among components within the same framework.

(3) *SafeInheritance_AmongClasses_WithInTheSameComponent()*: as explained in (5–25), this sub–predicate is used to ensure that inheritance is safe whenever used among components within the same framework.

(4) *SafeInheritance_AmongClasses_WithInTheSameSharedPool()*: as explained in (5–26), this sub–predicate is used to ensure that inheritance is safe whenever used among shared classes within the same shared pool.

$$SafeInheritance() \equiv \begin{bmatrix} InheritanceMayNotExistAcrossFrameworks() \wedge \\ SafeInheritance\_AmongComponents\_WithInTheSameFramework() \wedge \\ SafeInheritance\_AmongClasses\_WithInTheSameComponent() \wedge \\ SafeInheritance\_AmongClasses\_WithInTheSameSharedPool() \end{bmatrix} \quad (5\text{–}20)$$

In formula (5–21), the *GoodAssociation()* sub–predicate is formed by taking the conjunction of the three sub–predicates that are listed next and defined in more details in section 5.2.3.4.

(1) *GoodAssociationOnTheComponentLevel()*: as explained in (5–27), this sub–predicate is used to ensure that association among components is acceptable.

(2) *GoodAssociationOnTheInternalClassesLevel()*: as explained in (5–28), this sub–predicate is used to ensure that association among internal classes within the same component is acceptable.

(3) *AssociationMayNotExistAcrossSharedPools()*: this this sub–predicate is used to ensure that association does not exist among classes from different shared pools. This design decision can keep frameworks independent from each other.

$$GoodAssociation() \equiv \begin{bmatrix} GoodAssociationOnTheComponentsLevel() \wedge \\ GoodAssociationOnTheInternalClassesLevel() \wedge \\ AssociationMayNotExistAmongSharedPools \end{bmatrix} \quad (5\text{–}21)$$

As mentioned earlier in this section, the *GoodUseOfOtherRelations*() sub-predicate is left for future expansion.

$$GoodUsOfOtherRelations() \equiv \text{This will be left for potential expansion in the future} \qquad (5\text{–}22)$$

### 5.2.3.3 Explanation of (5–20): *SafeInheritance*()

In formula (5–23), predicate logic is used to specify our design decision for prohibiting components to inherit from each other across frameworks. This decision has been already justified in section 5.2.3.2.

Let $P1 \equiv \left((Component_{ij} \in Framework_i) \wedge \left(Inherits(Component_{ij}, Component_{kl})\right)\right)$, and
$P2 \equiv ((k = i) \wedge (Component_{kl} \in Framework_i))$, then
$InheritanceMayNotExistAcrossFrameworks() \equiv$

$$\left(\forall i,k, \begin{pmatrix}(1 \le i \le F) \wedge \\ (1 \le k \le F)\end{pmatrix} \Rightarrow \left(\forall j,l, \begin{pmatrix}(1 \le j \le C_i) \wedge \\ (1 \le l \le C_i)\end{pmatrix} \Rightarrow (P1 \Rightarrow P2)\right)\right) \qquad (5\text{–}23)$$

In formula (5–24), the *SafeInheritance_AmongComponents_WithInTheSameFramework*() sub-predicate of (5–20) above is formed by taking the conjunction of the two sub-predicates that are listed next and defined in more details in section 5.2.3.5.

(1) *A_ComponentMayInheritFromOnlyOneOtherComponentsFromTheSameFramework*(): as explained in (5–30), this sub-predicate is used to ensure that inheritance among components within the same framework can be only single (i.e. multiple inheritance is prohibited). It has been already justified in section 2.4.1.2 that multiple inheritance causes more complications to CO–Toolkits as well as to the applications that depend on components from these CO–Toolkits.

(2) *OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework*(): as explained in (5–31), this sub-predicate states that only public inheritance may

exist among components within the same framework. By this, the protected and private inheritance (i.e. allowed in C++) are not allowed to exist among classes that belong to different components, even within the same framework. Protected and private inheritance add external stuff to the protected and private sections of the inhering classes (i.e. affect the implementation of the inheriting classes). However, we do permit protected and private inheritance among classes given that both the inheriting and the inherited from classes belong to the same component.

$$SafeInheritance\_AmongComponents\_WithInTheSameFramework() \equiv \begin{pmatrix} A\_ComponentMayInheritFromOnlyOneOtherComponentsFromTheSameFramework() \wedge \\ OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework() \end{pmatrix} \quad (5\text{–}24)$$

In formula (5–25), the *SafeInheritance_AmongComponents_WithInTheSameComponent()* sub-predicate of (5–20) above is formed by taking the conjunction of the two sub-predicates that are listed next and defined in more details in section 5.2.3.6.

(1) *NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent()*: as explained in (5–32), this sub-predicate is used to ensure that inheritance among internal classes within the same component can be only single (i.e. multiple inheritance is prohibited). The justification for this decision is similar to the justification for permitting only single inheritance at the components level mentioned above.

(2) *InheritanceMayBe_Public/Protected/or/Private_OnlyWithInTheSameComponent()*: as explained in (5–31), this sub-predicate states that inheritance among internal classes within the same component could be public, protected, or private. The justification for this is that there is no danger from permitting this as long as these internal classes belong to the same component. That means, the way in-

ternal classes collaborate with each other is an internal design issue and that will not be exposed to the externals.

$$\mathit{SafeInheritance\_AmongClasses\_WithInTheSameComponent} \equiv \left( \begin{array}{l} \mathit{NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent}() \wedge \\ \mathit{InheritanceMayBe\_Public/Protected/or/Private\_OnlyWithInTheSameComponent}() \end{array} \right) \quad (5\text{–}25)$$

In formula (5–26), the *SafeInheritance_AmongClasses_WithInTheSameSharedPool()* sub–predicate of (5–20) above is formed by taking the conjunction of the two sub–predicates that are listed next and defined in more details in section 5.2.3.7.

(1) *InheritanceMayNotExistAcrossSharedPools()*: as explained in (5–34), this sub–predicate states that inheritance may not exist among classes from different shared pools. This is similar to the previous decision for prohibiting inheritance across frameworks.

(2) *NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool()*: as explained in (5–35), this sub–predicate states that multiple inheritance is prohibited among classes within the same shared pool. Again this similar to the previous decision for prohibiting multiple inheritance among components within the same framework as well as among internal classes within the same component.

$$\mathit{SafeInheritance\_AmongClasses\_WithInTheSharedPools}() \equiv \left( \begin{array}{l} \mathit{InheritanceMayNotExistAcrossSharedPools}() \wedge \\ \mathit{NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool}() \end{array} \right) \quad (5\text{–}26)$$

### 5.2.3.4 Explanation of (5–21): *GoodAssociation()*

In formula (5–27), the *GoodAssociationOnTheComponentsLevel()* sub–predicate of (5–21) above is formed by taking the conjunction of the two sub–predicates that are listed next and defined in more details in section 5.2.3.8.

(1) *ComponentsMayHaveOnlyUniDirectionalAssociationWithClassesFromSharedPools*(): as explained in (5–36), this sub–predicate states that only uni–directional association from components to shared–pools' classes. This means that a component may contain an object of a shared class. However, a shared class may not contain an object of any component's internal class since it makes no sense to permit this.

(2) *AssociationMayNotExistAcrossFrameworks*():as explain in (5–37), this this sub–predicate is used to ensure that association does not exist among components from different frameworks. This design decision can keep frameworks independent from each other.

$$GoodAssociationOnTheComponentsLevel() \equiv \left(\begin{array}{l} ComponentsMayHaveOnlyUniDirectionalAssociationWithClassesFromTheSharedPools() \wedge \\ AssociationMayNotExistAcrossFrameworks() \end{array}\right) \tag{5–27}$$

In formula (5–28), the *GoodAssociationOnTheInternalClassesLevel*() sub–predicate of (5–21) above is represented by a single sub–predicate. The justification for not replacing this sub–predicate by its final mathematical representation in (5–38) is that we want to keep this view open for any new rule that may be found in further research about using association among internal classes.

(1) *AssociationAndInheritanceMayNotExistBetweenTwoInternalClassesAtTheSameTime*(): as explained in (5–38), this sub–predicate states that both association and inheritance relations may not exist at the same time between the same two internal classes. This has been already justified in section 5.2.3.4.

$$GoodAssociationOnTheInternalClassesLevel() \equiv (AssociationAndInheritanceMayNotExistBetweenTwoClassesAtTheSameTime()) \tag{5–28}$$

In formula (5–29), predicate logic is used to specify our design decision for prohibiting components to associate with each other across frameworks.

Let $P1 \equiv \left(\left(\mathit{Component}_{ij} \in \mathit{Framework}_i\right) \wedge \left(\mathit{Associates}\left(\mathit{Component}_{ij}, \mathit{Component}_{kl}\right)\right)\right)$, and $P2 \equiv \left((k = i) \wedge (\mathit{Component}_{kl} \in \mathit{Framework}_i)\right)$, then

$\mathit{InheritanceMayNotExistAcrossFrameworks}() \equiv$

$$\left(\forall i,k,\begin{pmatrix}(1 \le i \le F) \wedge \\ (1 \le k \le F)\end{pmatrix} \Rightarrow \left(\forall j,l,\begin{pmatrix}(1 \le j \le C_i) \wedge \\ (1 \le l \le C_i)\end{pmatrix} \Rightarrow (P1 \Rightarrow P2)\right)\right) \tag{5–29}$$

### 5.2.3.5 Explanation of (5–24): *SafeInheritance_AmongComponents_WithInTheSameFramework()*

In formula (5–30), the definition for the first predicate of (5–24) above is final. We used the macros ***P1***, ***P2***, ***P3***, and ***P4*** just to simplify this predicate and improve its readability.

Let $P1 \equiv \begin{pmatrix}(1 \le j \le C_i) \wedge \\ \left(\mathit{Component}_{ij} \in \mathit{Framework}_i\right)\end{pmatrix}$, $P2 \equiv \left[\exists k, \left|\begin{bmatrix}(1 \le k \le C_i) \wedge (k \ne j) \wedge \\ \left(\mathit{Component}_{ik} \in \mathit{Framework}_i\right) \wedge \\ \left(\mathit{Inherits}\left(\mathit{Component}_{ij}, \mathit{Component}_{ik}\right)\right)\end{bmatrix}\right.\right]$,

$P3 \equiv \begin{pmatrix}(1 \le l \le C_i) \wedge (l \ne j) \wedge (l \ne k) \wedge \\ \left(\mathit{Component}_{il} \in \mathit{Framework}_i\right)\end{pmatrix}$, and $P4 \equiv \left(\mathit{Inherits}\left(\mathit{Component}_{ij}, \mathit{Component}_{il}\right)\right)$, then

$\mathit{A\_ComponentMayInheritFromOnlyOneOtherComponentFromTheSameFramework}() \equiv$

$$(\forall i, (1 \le i \le F) \Rightarrow (\forall j, (P1 \wedge P2) \Rightarrow (\forall l, P3 \Rightarrow \neg P4))) \tag{5–30}$$

In formula (5–31), the definition for the second predicate of (5–24) above is final too. Similar to (5–30), we used the macros ***P1***, and ***P2*** just to simplify this predicate and improve its readability.

Now let $P1 \equiv \begin{bmatrix} (1 \le j \le C_i) \wedge \\ (1 \le k \le C_i) \wedge (k \ne j) \wedge \\ (Component_{ij} \in Framework_i) \wedge \\ (Component_{ik} \in Framework_i) \wedge \\ (Inherits(Component_{ij}, Component_{ik})) \end{bmatrix}$, and $P2 \equiv (InheritanceCategory(Component_{ij}, Component_{ik}))$,

then

$$OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework() \equiv (\forall i, (1 \le i \le F) \Rightarrow (\forall j, k, (P1 \Rightarrow P2)) \quad (5\text{–}31)$$

### 5.2.3.6 Explanation of (5–25): *SafeInheritance_AmongClasses_WithInTheSameComponent()*

In formula (5–32), the definition for the first predicate of (5–25) above is final. We used the macros ***P1***, ***P2***, ***P3***, ***P4***, and ***P5*** just to simplify this predicate and improve its readability.

Let $P1 \equiv \begin{pmatrix} (1 \le j \le C_i) \wedge \\ (Component_{ij} \in Framework_i) \end{pmatrix}$, $P2 \equiv \begin{bmatrix} (1 \le l \le c_{ij}) \wedge \\ (InternalClass_{ijl} \in InternalClassesSet_{ij}) \end{bmatrix}$,

$P3 \equiv \left[ \exists k, \mid \begin{bmatrix} (1 \le k \le c_{ij}) \wedge \\ (k \ne l) \wedge \\ (InternalClass_{ijk} \in InternalClassesSet_{ij}) \wedge \\ (Inherits(InternalClass_{ijl}, InternalClass_{ijk})) \end{bmatrix} \right]$,

$P4 \equiv \begin{bmatrix} (1 \le f \le c_{ij}) \wedge \\ (f \ne k) \wedge (f \ne l) \wedge \\ (InternalClass_{ijf} \in InternalClassesSet_{ij}) \end{bmatrix}$, and $P5 \equiv (Inherits(Class_{ijl}, Class_{ijf}))$ then

$$NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent() \equiv (\forall i (1 \le i \le F) \Rightarrow (\forall j, P1 \Rightarrow (\forall l, (P2 \wedge P3) \Rightarrow (P4 \Rightarrow \neg P5)))) \quad (5\text{–}32)$$

In formula (5–33), the definition for the second predicate of (5–25) above is final too. We used the macros ***P1***, and ***P2*** just to simplify this predicate and improve its readability.

Now let $\boldsymbol{P1} \equiv \begin{bmatrix} (1 \le j \le C_i) \wedge \\ (1 \le k \le C_i) \wedge \\ (k \ne j) \end{bmatrix}$, $\boldsymbol{P2} \equiv \begin{bmatrix} (\mathit{InternalClass}_{ij} \in \mathit{InternalClassesSet}_{ij}) \wedge \\ (\mathit{InternalClass}_{ik} \in \mathit{InternalClassesSet}_{ij}) \wedge \\ (\mathit{Inherits}(\mathit{InternalClass}_{ij}, \mathit{InternalClass}_{ik})) \end{bmatrix}$, and

$\boldsymbol{P3} \equiv \left[ \mathit{InheritanceCategory}(\mathit{InternalClass}_{ijl}, \mathit{InternalClass}_{ik}) \in \left\{ \begin{matrix} \text{'public'}, \\ \text{'protected'}, \\ \text{'private'} \end{matrix} \right\} \right]$, then

$$\mathit{InheritanceMayBe\_Public/Protected/or/Private\_OnlyWithInTheSameComponent}() \equiv$$

$$\left(\forall i, (1 \le i \le F) \Rightarrow \left(\forall j, k, \boldsymbol{P1} \Rightarrow \left(\forall l, \left(1 \le l \le c_{ij}\right) \Rightarrow (\boldsymbol{P2} \Rightarrow \boldsymbol{P3})\right)\right)\right) \qquad (5\text{–}33)$$

### 5.2.3.7 Explanation of (5–26): *SafeInheritance_AmongClasses_WithInTheSharedPools()*

In formula (5–34), the definition for the first predicate of (5–26) above is final. We used the macros ***P1***, ***P2***, and ***P3*** just to simplify this predicate and improve its readability.

Let $\boldsymbol{P1} \equiv \begin{bmatrix} (\mathit{SharedClass}_{ij} \in \mathit{SharedClassesSet}_i) \wedge \\ (\mathit{SharedClass}_{kl} \in \mathit{SharedClassesSet}_k) \wedge \\ (\mathit{Inherits}(\mathit{SharedClass}_{ij}, \mathit{SharedClass}_{kl})) \end{bmatrix}$, and $\boldsymbol{P2} \equiv \begin{pmatrix} (i = k) \wedge (j \ne l) \wedge \\ (\mathit{SharedPool}_i = \mathit{SharedPool}_k) \end{pmatrix}$, then

$$\mathit{InheritanceMayNotExistAcrossSharedPools}() \equiv$$

$$\left[\forall i, k, \begin{pmatrix} (1 \le i \le S) \wedge \\ (1 \le k \le S) \end{pmatrix} \Rightarrow \left[\forall j, l, \begin{pmatrix} (1 \le j \le c_i^s) \wedge \\ (1 \le l \le c_k^s) \end{pmatrix} \Rightarrow (\boldsymbol{P1} \Rightarrow \boldsymbol{P2})\right]\right] \qquad (5\text{–}34)$$

In formula (5–35), the definition for the second predicate of (5–26) above is final too. We used the macros ***P1***, ***P2***, and ***P3*** just to simplify this predicate and improve its readability.

Now let $\boldsymbol{P1} \equiv \left[ \exists k, \middle| \begin{bmatrix} (1 \le k \le c_i^s) \wedge (k \ne j) \wedge \\ (\mathit{SharedClass}_{ik} \in \mathit{SharedClassesSet}_i) \wedge \\ (\mathit{SharedClass}_{ij} \in \mathit{SharedClassesSet}_i) \wedge \\ (\mathit{Inherits}(\mathit{SharedClass}_{ij}, \mathit{SharedClass}_{ik})) \end{bmatrix} \right]$,

$\boldsymbol{P2} \equiv (\mathit{SharedClass}_{il} \in \mathit{SharedClassesSet}_i)$, and $\boldsymbol{P3} \equiv (\mathit{Inherits}(\mathit{SharedClass}_{ij}, \mathit{SharedClass}_{il}))$, then

$$NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool() \equiv$$

$$\left[\forall i, (1 \le i \le S) \Rightarrow \left[\forall j, \left(1 \le j \le c_i^s\right) \Rightarrow \left[P1 \Rightarrow \left[\forall f, \begin{bmatrix} \left(1 \le f \le c_i^s\right) \wedge \\ (f \ne j) \wedge \\ (f \ne k) \end{bmatrix} \Rightarrow (P2 \Rightarrow \neg P3)\right]\right]\right]\right] \quad (5\text{–}35)$$

## 5.2.3.8 Explanation of (5–27): *GoodAssociationOnTheComponentLevel()*

In formula (5–36), the definition for the first predicate of (5–27) above is final. We used the macros ***P1***, and ***P2*** just to simplify this predicate and improve its readability.

$$\text{Let } P1 \equiv \begin{bmatrix} \left(Component_{ij} \in Framework_i\right) \wedge \\ \left(InternalClass_{ijl} \in InternalClassesSet_{ij}\right) \wedge \\ \left(SharedClass_{kf} \in SharedClassesSet_k\right) \wedge \\ \left(Associates\left(InternalClass_{ijl}, SharedClass_{kf}\right)\right) \end{bmatrix}, \text{ and}$$

$$P2 \equiv \begin{bmatrix} \left(Fr\_Name(Framework_i) \in Frs\_PermittedToAccess\_SP(k)\right) \wedge \\ \left(One\_to\_One\_UniDirectionalAssociation\left(Class_{ijl}, SharedClass_{kf}\right)\right) \wedge \\ \left(\neg\left(BiDirectionalAssociation\left(Class_{ijl}, SharedClass_{kf}\right)\right)\right) \end{bmatrix}, \text{ then}$$

$$ComponentsMayHaveOnlyUnidirectionalAssociationWithClassesFromTheSharedPools() \equiv$$

$$\left[\forall i, k, \left(\begin{matrix} (1 \le i \le F) \wedge \\ (1 \le k \le S) \end{matrix}\right) \Rightarrow \left[\forall j, l, f, \begin{bmatrix} (1 \le j \le C_i) \wedge \\ \left(1 \le l \le c_{ij}\right) \wedge \\ \left(1 \le f \le c_k^s\right) \end{bmatrix} \Rightarrow (P1 \Rightarrow P2)\right]\right] \quad (5\text{–}36)$$

In formula (5–37), the definition for the first predicate of (5–27) above is final too. We used the macros ***P1***, and ***P2*** just to simplify this predicate and improve its readability.

$$\text{Now let } P1 \equiv \left(\begin{matrix} \left(Component_{ij} \in Framework_i\right) \wedge \\ \left(Component_{kl} \in Framework_k\right) \end{matrix}\right), \text{ and } P2 \equiv \left(Associates\left(Component_{ij}, Component_{kl}\right)\right), \text{ then}$$

$$AssociationMayNotExistAcrossFrameworks() \equiv$$

$$\left[\forall i, k, \begin{bmatrix} (1 \le i \le F) \wedge \\ (1 \le k \le F) \wedge \\ (i \ne k) \end{bmatrix} \Rightarrow \left(\forall j, l, \left(\begin{matrix} (1 \le j \le C_i) \wedge \\ (1 \le l \le C_k) \end{matrix}\right) \Rightarrow (P1 \Rightarrow \neg P2)\right)\right] \quad (5\text{–}37)$$

### 5.2.3.9 Explanation of (5–28): *GoodAssociationOnTheInternalClassesLevel()*

In formula (5–38), the definition for the first predicate of (5–28) above is final too. We used the macros ***P1***, and ***P2*** just to simplify this predicate and improve its readability.

$$\text{Let } \boldsymbol{P1} \equiv \begin{bmatrix} (\boldsymbol{Component_{ij}} \in \boldsymbol{Framework_i}) \wedge \\ (\boldsymbol{Component_{ik}} \in \boldsymbol{Framework_i}) \wedge \\ (\boldsymbol{InternalClass_{ijl}} \in \boldsymbol{InternalClassesSet_{ij}}) \wedge \\ (\boldsymbol{InternalClass_{ikf}} \in \boldsymbol{InternalClassesSet_{ik}}) \end{bmatrix}, \text{ and}$$

$$\boldsymbol{P2} \equiv \begin{bmatrix} (\boldsymbol{Inherits}(\boldsymbol{InternlaClass_{ijl}}, \boldsymbol{InternalClass_{ikf}})) \wedge \\ (\boldsymbol{Associates}(\boldsymbol{InternalClass_{ijl}}, \boldsymbol{InternalClass_{ikf}})) \end{bmatrix}, \text{ then}$$

$$AssociationAndInheritanceMayNotExistBetweenTwoClassesAtTheSameTime() \equiv$$

$$\left[\forall i, (1 \le i \le F) \Rightarrow \left[\forall j, k, \begin{pmatrix} (1 \le j \le C_i) \wedge \\ (1 \le k \le C_i) \end{pmatrix} \Rightarrow \left[\forall l, f, \begin{bmatrix} (1 \le l \le c_{ij}) \wedge \\ (1 \le f \le c_{ik}) \wedge \\ ((j = k) \Rightarrow (l \ne f)) \end{bmatrix} \Rightarrow (P1 \Rightarrow \neg P2)\right]\right]\right] \quad (5\text{–}38)$$

## 5.2.4 The View Lexicon

In this sub–section, we clarify all variables, symbols, acronyms, constants, and auxiliary functions that appear in the formalism of the above view and its characteristics.

### 5.2.4.1 Variables And Acronyms

***CO***: Component Oriented.

***COToolkit***: Represents any Component–Oriented Toolkit that can provide reusable components to be used to build new applications.

$\boldsymbol{Component_{ij}}$: Represents the $j^{th}$ component in the $i^{th}$ framework.

$\boldsymbol{Framework_i}$: Represents the $i^{th}$ framework.

$\boldsymbol{Implementation_{ij}}$: Represents the implementation of the $ij^{th}$ component.

$\boldsymbol{InheritedInterface_{ij}}$: Represents the inherited interface of the $ij^{th}$ component. This refers to the interface of the methods that are inherited by that component.

***Interface***$_{ij}$: Represents the interface of the $ij^{th}$ component.

***InternalClassesSet***$_{ij}$: Represents the set of the internal classes in the $ij^{th}$component.

***InternalClass***$_{ijl}$: Represents the implementation of the $l^{th}$ internal class in the $ij^{th}$component.

***LocalInterface***$_{ij}$: Represents the local interface of the $ij^{th}$ component. This refers to the interface of the local non-inherited methods that belong to that component.

***PublicMethod***$_{ijlk}$: Represents the implementation of the $k^{th}$ public method of the $l^{th}$ internal class in the $ij^{th}$component.

***LocalPublicMethodSignature***$_{ijlk}$: Represents the signature (interface description) of the $k^{th}$ local method in the $l^{th}$ class that belongs to the $ij^{th}$ component.

***MainPart***$_{ij}$: Represents the implementation of the main part of the $ij^{th}$ component.

***OtherSharedEntitiesSet***$_{k}$: Represents the set of other shared entities in the $k^{th}$ shared-pool.

***OtherSharedEntity***$_{kf}$: Represents the $f^{th}$ other shared entity in the $k^{th}$ shared-pool.

***PubliclyInheritedMethodSignature***$_{ijlf}$: Represents the signature (interface description) of the $f^{th}$ method that is inherited by the $l^{th}$ class of the $ij^{th}$ component.

***PrivateMethod***$_{ijlf}$: Represents the implementation of the $f^{th}$ private method of he $l^{th}$ internal class in the $ij^{th}$component.

***PrivateObject***$_{ijlk}$: Represents the $k^{th}$ private object of the $l^{th}$ internal class in the $ij^{th}$ component.

***PrivateSection***$_{ijl}$: Represents the implementation of the private section of the $l^{th}$ internal class of the $ij^{th}$component.

***ProtectedObject***$_{ijlk}$: Represents the $k^{th}$ protected object of the $l^{th}$ internal class of the $ij^{th}$ component.

***ProtectedMethod***$_{ijlf}$: Represents the implementation of the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$component.

***ProtectedSection***$_{ijl}$: Represents the implementation of the protected section of the $l^{th}$ internal class in the $ij^{th}$component.

***PublicSection***$_{ijl}$: Represents the implementation of the public section of the $l^{th}$ internal class in the $ij^{th}$component.

***P_INs***$_{ijlk}$: Represents the sequence of inputs to the $k^{th}$ public local method of the $l^{th}$ internal class in the $ij^{th}$ component.

***P_OUTs***$_{ijlk}$: Represents the sequence of outputs from the $k^{th}$ public local method in the $l^{th}$ internal class of the $ij^{th}$ component.

$\textit{P\_ReturnedValue}_{ijlk}$: Represents the returned value by the $k^{th}$ public local method of the $l^{th}$ internal class in the $ij^{th}$ component.

$P_{Inhr}\textit{\_INs}_{ijlf}$: Represents the sequence of inputs to the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$P_{Inhr}\textit{\_OUTs}_{ijlf}$: Represents the sequence of outputs from the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$P_{Inhr}\textit{\_ReturnedValue}_{ijlf}$: Represents the returned value by the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{R\_INs}_{ijlf}$: Represents the sequence of inputs to the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{R\_OUTs}_{ijlf}$: Represents the sequence of outputs from the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{R\_ReturnedValue}_{ijlf}$: Represents the returned value by the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{SharedPool}_k$: Represents the $k^{th}$ shared-pool.

$\textit{SharedClass}_{kj}$: Represents the $j^{th}$ class in the $k^{th}$ shared-pool.

$SOF_X(A, B)$: Represents the implementation of method X.

$\textit{V\_INs}_{ijlf}$: Represents the sequence of inputs to the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{V\_OUTs}_{ijlf}$: Represents the sequence of outputs from the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\textit{V\_ReturnedValue}_{ijlf}$: Represents the returned value by the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

#### 5.2.4.2 Symbols And Constants

$F$: Represents the number of frameworks.

$S$: Represents the number of shared-pools.

$C_i$: Represents the number of components in the $i^{th}$ framework.

$c_{ij}$: Represents the number of local classes used to construct the $ij^{th}$ component.

$c_k^s$: Represents the number of shared classes in the $k^{th}$ shared-pool.

$e_k^s$: Represents the number of other shared entities in the $k^{th}$ shared-pool.

$M^{P}_{ijl}$: Represents the number of local public methods in $l^{th}$ class of the $ij^{th}$ component.

$pm^{IN}_{ijlk}$: Represents the number of inputs of the $k^{th}$ local public method in the $l^{th}$ class of the $ij^{th}$ component.

$pm^{OUT}_{ijlk}$: Represents the number of outputs of the $k^{th}$ local public method in the $l^{th}$class of the $ij^{th}$ component.

$M^{P_{Inhr}}_{ijl}$: Represents the number of public methods inherited by the $l^{th}$ class of the $ij^{th}$ component.

$pm^{IN_{Inhr}}_{ijlf}$: Represents the number of inputs of the $f^{th}$ public method that is inherited by the $l^{th}$ class of the $ij^{th}$ component.

$pm^{OUT_{Inhr}}_{ijlf}$: Represents the number of outputs of the $f^{th}$ public method that is inherited by the $l^{th}$ class of the $ij^{th}$component.

$O^{R}_{ijl}$: Represents the number of objects in the protected section of the $l^{th}$ class of the $ij^{th}$ component.

$M^{R}_{ijl}$: Represents the number of local protected methods in the $l^{th}$ class of the $ij^{th}$ component.

$rm^{IN}_{ijlf}$: Represents the number of inputs of the $f^{th}$ local protected method in the $l^{th}$ class of the $ij^{th}$component.

$rm^{OUT}_{ijlf}$: Represents the number of outputs of the $f^{th}$ local protected method in the $l^{th}$class of the $ij^{th}$component.

$O^{V}_{ijl}$: Represents the number of objects in the private section of the $l^{th}$ class of the $ij^{th}$ component.

$M^{V}_{ijl}$: Represents the number of local private methods in the $l^{th}$ class of the $ij^{th}$ component.

$vm^{IN}_{ijlf}$: Represents the number of inputs of the $f^{th}$ local private method in the $l^{th}$ class of the $ij^{th}$ component.

$vm^{OUT}_{ijlf}$: Represents the number of outputs of the $f^{th}$ local private method in the $l^{th}$class of the $ij^{th}$ component.

### 5.2.4.3 Auxiliary Functions

***Frs_PermittedToAccessSP(k)***: An auxiliary function that returns the set of frameworks that are permitted to access the $k^{th}$ shared-pool.

***One_to_One_UniDirectionalAssociation(X, Y)*** ≡ An auxiliary function that returns TRUE if Class X contains an object of Class Y while Y does not contain any object of X.

***One_to_One_BiDirectionalAssociation(X, Y)*** ≡ An auxiliary function that returns TRUE if Classes X and Y each contains an object of the other.

***One_to_Many_UniDirectionalAssociation(X, Y)*** ≡ An auxiliary function that returns TRUE if Class X contains a set of objects of Class Y while Y does not contain any object of X.

***One_to_Many_BiDirectionalAssociation(X, Y)*** $\equiv$ An auxiliary function that returns TRUE if Class X contains a set of objects of Class Y while Y contains only one object of class X.

***Many_to_Many_BiDirectionalAssociation(X, Y)*** $\equiv$ An auxiliary function that returns TRUE if Classes X and Y each contains multiple objects of the other.

$$\textit{UniDirectionalAssociation}(X, Y) \equiv \begin{pmatrix} \textit{One\_to\_One\_UniDirectionalAssociation}(X, Y) \vee \\ \textit{One\_to\_Many\_UniDirectionalAssociation}(X, Y) \end{pmatrix}$$

: An auxiliary function that returns TRUE if X and Y have a form of uni–directional association with each other.

$$\textit{BiDirectionalAssociation}(X, Y) \equiv \begin{bmatrix} \textit{One\_to\_One\_BiDirectionalAssociation}(X, Y) \vee \\ \textit{One\_to\_Many\_BiDirectionalAssociation}(X, Y) \vee \\ \textit{Many\_to\_Many\_BiDirectionalAssociation}(X, Y) \end{bmatrix}$$

: An auxiliary function that returns TRUE if X and Y have a form of bi–directional association with each other.

$$\textit{Associates}(X, Y) \equiv \begin{pmatrix} \textit{UniDirectionalAssociation}(X, Y) \vee \\ \textit{BiDirectionalAssociation}(X, Y) \end{pmatrix}$$

: An auxiliary function that returns TRUE if classes X and Y associate with each other.

***InheritanceCategory(Inherits(X, Y))***: An auxiliary function that returns the inheritance category when component X inherits from component Y. The returned value is either "Public", "Protected", or "Private".

***Inherits(X, Y)***: An auxiliary boolean function that returns TRUE if component X does inherit from component Y. Otherwise, it returns FALSE.

***Type(X)***: An auxiliary function that returns the data type of the argument *X*.

## 5.3 Conclusions

In this chapter, we have shown the following:

(1) Rules and guidelines for CO–Toolkits' design and implementation structures can be formalized using a convenient mathematical terminology and notation (*c.f. section 5.2*).

(2) Such formalized rules for design and implementation structures can be used to build a checking tool. An example prototype tool that we built based on these rules is described in Appendix B.

(3) Such checking tools can be used to check CO–Toolkits' design and implementation structures and report any rule violations back to the developers of

these CO–Toolkits. Examples are provided in Appendixes B2–B4. In particular, an example of ACE inheritance structure is presented in Appendix B4.

# CHAPTER 6

# THIRD VIEW: FORMALIZING HOW TO DOCUMENT CO–TOOLKITS

## 6.1 Overview

This chapter describes our documentation view for CO–Toolkits. Documents that are provided with CO–Toolkits are hard to understand. For example, the documents of ACE [43–52] are a set of papers that can help those who are already familiar with ACE. Besides the papers, a lot of comment lines are inserted in the source code of ACE. In such cases, developers may have to start reading through the code itself in order to understand the CO–Toolkit that is under investigation.

We believe that proper documents can help developers understand the CO–Toolkits they are looking at without having to navigate throughout their code. In this chapter, we want to show what kinds of proper documents we think should be provided for CO–Toolkits, what contents should be included in these documents, and how they should be organized. The main documents would be a *Component Guide Document (CGD)*, a *Component Interface Specifications Document (CISD)*, and a *Component Internal Design Specifications Document (CIDSD)*. We expect such documents to maintain the basic properties of the formal documents produced by the Software Engineering Research

Group at McMaster University [4, 7, 36–40, 53]. Appendix C provides sample tables for this third view.

## 6.2 Component Guide Document

A Component Guide Document (CGD) of a given CO–Toolkit is intended to provide an overview of how files are structured, how components are grouped into frameworks, and how these components are dependent on each other (i.e. the relations among these components such as inheritance and association). In summary, a CGD is an informal document that can be read first. This can help to understand the other provided documents (i.e. the interface specifications and internal design specifications ones).

### 6.2.1 What Information Should Be Contained In A "Component Guide Document"? And How Should It Be Organized?

As mentioned above, CGDs are informal documents that contain pictorial and textual contexts. Briefly, a CGD of a given CO–Toolkit is expected to contain the following chapters:

(1) A brief description for the given CO–Toolkit and its frameworks and their components.

(2) A brief description for each document that is provided with the given CO–Toolkit.

(3) A brief description associated with a picture that illustrates the structuring of frameworks in the given CO–Toolkit.

(4) A brief description associated with a set of pictures that illustrate the file structures implemented in the given CO–Toolkit. Each picture is expected to illustrate the file structure that is implemented in one framework.

(5) A brief description associated with a set of pictures that illustrate the relations among components of that CO–Toolkit. Each picture is expected to illustrate the inheritance and association relations among the components of the same framework as well as the relations among these components and other classes from the shared pools.

(6) A table that can serve as quick reference for what components are provided by each framework, and what files are used to implement each of these components. Readers can refer to appendix C for a guiding example.

(7) A table that can serve as a quick reference for what services each component can provide. CGDs can be helpful to those who may be interested in CO–Toolkits (i.e. maintainers, users, etc.). By looking at the service–indexed table, a user can easily find out if the services he needs are in that table. If those services are found, then he can easily locate the needed components.

### 6.2.2 Who Is A "Component Guide Document" intended For?

A CGD of a given CO–Toolkit is intended for maintainers, users, sales, and any other people who are interested in having a quick overview of that CO–Toolkit.

#### 6.2.2.1 Maintainers

A CGD of a given CO–Toolkit can help maintainers to better understand that toolkit. For example, the pictorial views of the file structure, and the relations among components provide a quick and clear overview of the given CO–Toolkit. This reminds us with the motto "*a picture is worth a thousand words*". Indexed tables tell maintainers which component provides which service. Also, such tables tell which file of code to look at when it comes to modify a certain component.

#### 6.2.2.2 Users (Developers of CO–Applications)

As already mentioned in this thesis, the word users refers to developers who are interested in integrating components from CO–Toolkits to build their CO–applications. To be able to use a CO–Toolkit, users need to have a clear understanding the conjunction of of that toolkit. We believe that the details provided in the CGD (in particular, the indexed table of components) can guide users to choose the components that can use to build their applications.

#### 6.2.2.3 Sales Staff

Sales staff can use some of the pictorial views that are provided in CGDs in their promotional material and advertisement. This requires such pictorial views to be well–organized and clearly presented in the CGD.

#### 6.2.2.4 Technical Support

Technical support staff can understand a given CO–Toolkit by first reading its CGD. Reading CGDs can help technical support staff to better deal with problems raised by customers (i.e. users of CO–Toolkits). In our text, technical support staff are somehow different from maintainers (but still one person can play both roles) since technical support staff are in the middle between maintainers and customers. They understand their customers' problems, duplicate these problems in house, and then pass them to maintainers. Maintainers in turn, fix the problems and pass their solutions back to customers through technical support staff.

### 6.2.3 When Should A "Component Guide Document" Be Read?

We think that a CO–Toolkit's CGD should be read first. This is because a CGD provides various informal information on the different aspects of the given CO–Toolkit.

For example, a CGD illustrates the file structure and the relations among components (i.e. inheritance, association, etc.) in both pictorial and textual forms of context. This can make it easier to understand the general features of the given CO–Toolkit. Also, this can help understand other provided documents with less pain.

## 6.3 Component Interface Specifications Document

A Component Interface Specifications Document (CISD) of a given CO–Toolkit is intended to provide a precise description (specification) for the interface of each component that is provided in that CO–Toolkit.

### 6.3.1 What Information Should Be Contained In A "Component Interface Specifications Document"? And How Should It Be Organized?

A Component Interface Specifications Document (CISD) can be organized into chapters such that each chapter provides the complete interface specifications for a particular framework from the given CO–Toolkit. Consequently, each of these chapters can be divided into sections such that each section provides the interface specifications for a particular component from that framework. An example for how to make interface specifications for components is provided in Appendix C12. We should acknowledge the fact that this kind of specifications is being investigated at SERG [53–54]. The interface specification for each component is divided into six parts (1) CHARACTERISTICS, (2) SYNTAX (3) CANONICAL REPRESENTATION, (4) EQUIVALENCE, (5) VALUES, and (6) LEXICON. For more details about these parts, readers can refer to Parnas and Mady's work presented in the " Functional Documentation for Computer Systems Engineering" paper [39].

### 6.3.2 Who Is A "Component Interface Specifications Document" intended For?

CISDs are mainly intended for users who are interested in building their applications from existing components. The interface specifications describes how to invoke these components.

Maintainers are obligated to keep CISDs updated. Whenever they make any modifications or changes to any piece of code of a given CO–Toolkit, it is essential that they reflect such modifications to all documents of that toolkit including the CISD. Otherwise, documents will loose their correctness and become useless.

Technical support staff need to read through CISDs. This can help them to understand the CO–Toolkits they are dealing with. This also can help them to reproduce problems reported by their customers.

### 6.3.3 When Should A "Component Interface Specifications Document" Be Read?

We suggest that a CO–Toolkit's CISD to be read after the CGD.

## 6.4 Component Internal Design Specifications Document

Similar to CISDs, a CO–Toolkit's CIDSD is intended to provide a precise description (specification) for the internal design of each component that is provided in that CO–Toolkit.

### 6.4.1 What Information Should Be Contained In A "Component Internal Design Specifications Document"? And How Should It Be Organized?

Similar to CISDs, CIDSDs can be organized into chapters such that each chapter provides the complete internal design specifications for a particular framework. Consequently, each chapter can be divided into sections such that each section provides the internal design specifications for a particular component. SERG kinds of specifications for internal designs [53–54] can still be used to provide the internal design specifications for components.

### 6.4.2 Who Is A "Component Internal Design Specifications Document" intended For?

CIDSDs are mainly intended for maintainers. They can use the CIDSD of the given CO–Toolkit to look at the internal design specifications for the component to be fixed. Such internal specifications can help them to locate the code portion that causes the problem. Again, we should mention that it is the obligation of maintainers to keep all documents up to date.

### 6.4.3 When Should A "Component Internal Design Specifications Document" Be Read?

Since CIDSDs are dealing with a low level of specifications that could be hard to understand, we suggest that a CO–Toolkit's CIDSD be read after the CGD and CISD of that CO–Toolkit.

## 6.5 Towards A Mathematical View For CO–Toolkits' Documentation

### 6.5.1 Purpose

This view is intended to formalize how to document CO–Toolkits. Also, it is intended to show how NRL documentation methods such those used by Parnas Group at McMaster University [36–40] can be used to document CO–Toolkits.

### 6.5.2 The View Structure

A CO–Toolkit documentation structure is represented by a tuple of that CO–Toolkit's documents. Readers can refer to the lexicon section of this view (*cf. section 6.5.4*) for the meanings of the variables and symbols used this section.

In formula (6–1), a CO–Toolkit is represented by a tuple of the following four elements:

(1) ***CGD***: this is used to represent the **Component Guide Document** as described in *section 6.5.2.1.*

(2) ***CISD***: this is used to represent the **Component Interface Specifications Document** as described in *section 6.5.2.2.*

(3) ***CIDSD***: this is used to represent the **Component Internal Design Specifications Document** as described in *section 6.5.2.3.*

(4) ***OtherDocuments***: this is used to accommodate any future expansion to this view whenever there is a need to provide more documents.

$$COToolkit \equiv \begin{bmatrix} CGD, \\ CISD, \\ CIDSD, \\ OtherDocuments \end{bmatrix} \tag{6–1}$$

### 6.5.2.1 Explanation of The CGD Document In (6–1) *COToolkit*

In formula (6–2), the *CGD* document of (6–1) above is represented by a tuple of the following three elements:

(1) ***CGD_DescriptiveOverview***: this element represents the first chapter of the proposed CGD document. This chapter should describe the various aspects of the CO–Toolkit under consideration. For example, it should describe the structure of the CO–Toolkit's frameworks as well as their components. Also, brief descriptions for the other provided document can be included in this chapter.

(2) ***CGD_PictorialOverview***: this element represents the second chapter of the proposed CGD document. This chapter should provide a set of pictorial views that describe several things such as the file structure used for the considered CO–Toolkit, the various relations across the frameworks and among their components (i.e. such as inheritance and association relations), and etc.

(3) ***CGD_IndexedQuickReferenceTables***: this element represents the third chapter of the proposed CGD document. This chapter should provide a set of index table that can help users as well as maintainers to find out what services are provided by each component.

$$CGD \equiv \begin{bmatrix} CGD\_DescriptiveOverview, \\ CGD\_PictorialOverview, \\ CGD\_IndexedQuickReferenceTables \end{bmatrix} \qquad (6\text{–}2)$$

### 6.5.2.2 Explanation of The CISD Document In (6–1) *COToolkit*

In formula (6–3), the ***CISD*** document of (6–1) above is represented by a sequence of chapters such that each chapter provides the interface specifications for a certain framework. As explained later on, each of these chapters will be divided into sections such that each section provides the interface specifications for a certain component.

Readers can refer to the lexicon of this view (*i.e. section 6.5.4*) for the meanings of the different variables and symbols used in this view. For example, $F$ is used to represent the number of frameworks provided in the considered CO–Toolkit.

$$CISD \equiv \langle CISD\_Chapter\_Framework_i \rangle_{i=1}^{F} \tag{6–3}$$

#### 6.5.2.2.1 Explanation of (6–3): *CISD*

In formula (6–4), the interface specifications of the $i^{th}$ framework (i.e. $CISD\_Chapter\_Framework_i$ of (6–3) above) is represented by a set of sections such that each section provides the interface specifications for a certain component (i.e. $CISD\_Section\_Component_{ij}$). As explained in the lexicon of this view (*i.e. section 6.5.4*), $C_i$ is used to represent the number of Components provided in the $i^{th}$ framework.

$$CISD\_Chapter\_Framework_i \equiv \langle CISD\_Section\_Component_{ij} \rangle_{j=1}^{C_i} \tag{6–4}$$

#### 6.5.2.2.2 Explanation of (6–4): $CISD\_Chapter\_Framework_i$

In formula (6–5), the interface specifications of the $ij^{th}$ component (i.e. $CISD\_Section\_Component_{ij}$ of (6–4) above) is represented by a tuple of the following six elements. We should notice that these elements are used for the same meanings as in Parnas's works [36–40].

(1) $CISD\_SubSection\_CHARACTERISTICS_{ij}$: as explained in (6–6), this element represents the sub–section that provides the general characteristics of the $ij^{th}$ component.

(2) $CISD\_SubSection\_SYNTAX_{ij}$: as explained in (6–7), this element represents the sub–section that provides the signatures of the access programs of the $ij^{th}$ component.

(3) $CISD\_SubSection\_CANONICAL\_REPRESENTATION_{ij}$: as explained in (6–8), this element represents the sub-section that provides the canonical representation for the $ij^{th}$ component.

(4) $CISD\_SubSection\_EQUIVALENCES_{ij}$: as explained in (6–9), this element represents the sub-section that provides the equivalences for the $ij^{th}$ component.

(5) $CISD\_SubSection\_VALUES_{ij}$: as explained in (6–10), this element represents the sub-section that provides the values returned by the access programs of the $ij^{th}$ component.

(6) $CISD\_SubSection\_LEXICON_{ij}$: this element represents the sub-section that provides the lexicon (i.e. dictionary) that provides the needed clarifications for the symbols, variables, etc. that are used in the interface specifications of the $ij^{th}$ component.

$$CISD\_Section\_Component_{ij} \equiv \begin{bmatrix} CISD\_SubSection\_CHARACTERISTICS_{ij}, \\ CISD\_SubSection\_SYNTAX_{ij}, \\ CISD\_SubSection\_CANONICAL\_REPRESENTATION_{ij}, \\ CISD\_SubSection\_EQUIVALENCES_{ij}, \\ CISD\_SubSection\_VALUES_{ij}, \\ CISD\_SubSection\_LEXICON_{ij} \end{bmatrix} \quad (6–5)$$

### 6.5.2.2.3 Explanation of (6–5): $CISD\_Section\_Component_{ij}$

In formula (6–6), the $CISD\_SubSection\_CHARACTERISTICS_{ij}$ of (6–5) above, is represented by a tuple of the following five elements:

(1) $TypeSpecified_{ij}$: this element represents the specification type of the $ij^{th}$ component.

(2) $PublicPrimitiveSystemDataTypes_{ij}$: this element represents the set of primitive data types (i.e. int, boolean, etc.) that appear in the interface specifications of the $ij^{th}$ component.

(3) $\boldsymbol{PublicConstantParameters_{ij}}$: this element represents the set of public constant that appear in the interface specifications of the $ij^{th}$ component.

(4) $\boldsymbol{PublicSpecificationTypes_{ij}}$: this element represents the set of types that appear in the interface specifications of the $ij^{th}$ component.

(5) $\boldsymbol{PublicSpecificationVariables_{ij}}$: this element represents the set of variables that appear in the interface specifications of the $ij^{th}$ component.

$$CISD\_SubSection\_CHARACTERISTICS_{ij} \equiv \begin{bmatrix} \boldsymbol{TypeSpecified_{ij}}, \\ \boldsymbol{PublicPrimitiveSystemDataTypes_{ij}}, \\ \boldsymbol{PublicConstantParameters_{ij}}, \\ \boldsymbol{PublicSpecificationTypes_{ij}}, \\ \boldsymbol{PublicSpecificationVariables_{ij}} \end{bmatrix} \tag{6–6}$$

In formula (6–7), the $\boldsymbol{CISD\_SubSection\_SYNTAX_{ij}}$ of (6–5) above, is represented by a tuple of the following two elements:

(1) $\boldsymbol{Public\_ACCESS\_ProgramsInComponent_{ij}}$: this element represents the set of public access programs (methods) that are contained in the $ij^{th}$ component.

(2) $\boldsymbol{InheritedPublic\_ACCESS\_ProgramsByComponent_{ij}}$: this element represents the set of access programs (methods) that are publicly inherited by the $ij^{th}$ component from other components.

$$CISD\_SubSection\_SYNTAX_{ij} \equiv \begin{bmatrix} \boldsymbol{Public\_ACCESS\_ProgramsInComponent_{ij}}, \\ \boldsymbol{InheritedPublic\_ACCESS\_ProgramsByComponent_{ij}} \end{bmatrix} \tag{6–7}$$

In formula (6–8), the $\boldsymbol{CISD\_SubSection\_CANONICAL\_REPRESENTATION_{ij}}$ of (6–5) above, is represented by a tuple of the following two elements:

(1) $\boldsymbol{CANONICAL\_REPRESENTATION\_OfComponent_{ij}}$: this element represents the canonical representation for the $ij^{th}$ component. As explained in (6–11), a canonical rep-

resentation of a component is composed of the canonical representations of its internal classes.

(2) $\mathit{ReferencesTo\_CANONICAL\_REPRESENTATIONS\_OFInheritedComponents}_{ij}$: this element represents a set of references to the canonical representations of the other components that are inherited from by the $ij^{th}$ component.

$$\mathit{CISD\_SubSection\_CANONOCAL\_REPRESENTATION}_{ij} \equiv \begin{bmatrix} \mathit{CANONOCAL\_REPRESENTATION\_OfComponent}_{ij}, \\ \mathit{ReferencesTo\_CANONICAL\_REPRESENTATIONS\_OfInheritedComponents}_{ij} \end{bmatrix} \quad (6\text{–}8)$$

In formula (6–9), the $\mathit{CISD\_SubSection\_EQUIVALENCES}_{ij}$ of (6–5) above, is represented by a tuple of the following two elements:

(1) $\mathit{EQUIVALENCES\_OfComponent}_{ij}$: this element represents the set of equivalent states to the canonical one. An object of the $ij^{th}$ component goes into certain one of these equivalent states whenever a certain access program is invoked on that object.

(2) $\mathit{ReferencesTo\_EQUIVALENCES\_OFInheritedComponents}_{ij}$: this element represents a set of references to the equivalences of the other components that are inherited from by the $ij^{th}$ component.

$$\mathit{CISD\_SubSection\_EQUIVALENCES}_{ij} \equiv \begin{bmatrix} \mathit{EQUIVALENCES\_OfComponent}_{ij}, \\ \mathit{ReferencesTo\_EQUIVALENCES\_OfInheritedComponents}_{ij} \end{bmatrix} \quad (6\text{–}9)$$

In formula (6–10), the $\mathit{CISD\_SubSection\_VALUES}_{ij}$ of (6–5) above, is represented by a tuple of the following two elements:

(1) $\mathit{VALUES\_OfComponent}_{ij}$: this element represents the set of values that are returned by the public access programs (methods) of the $ij^{th}$ component.

(2) ***ReferencesTo_VALUES_OFInheritedComponents**$_{ij}$*: this element represents a set of references to the values returned by the public access programs (methods) of the other components that are inherited from by the $ij^{th}$ component.

$$CISD\_SubSection\_VALUES_{ij} \equiv \begin{bmatrix} VALUES\_OfComponent_{ij}, \\ ReferencesTo\_VALUES\_OfInheritedACCESS\_Programs_{ij} \end{bmatrix} \qquad (6\text{–}10)$$

#### 6.5.2.2.4 Explanation of (6–8): *CISD_SubSection_CANONICAL_REPRESENTATION*$_{ij}$

Each component's CANONICAL representation is specified by the sequence of the CANONICAL representations of its internal classes.

$$CANONICAL\_REPRESENTATION\_OfComponent_{ij} \equiv \left\langle CANONICAL\_REPRESENTATION\_OfInternalClass_{ijl} \right\rangle_{l=1}^{c_{ij}} \qquad (6\text{–}11)$$

### 6.5.2.3 Explanation of The CIDSD Document In (6–1) *COToolkit*

In formula (6–12), the *CIDSD* document of (6–1) above is represented by a tuple of the following two sequences of chapters:

(1) $\langle CIDSD\_Chapter\_Framework_i \rangle_{i=1}^{r}$: this element represents the sequence of chapters that provide the frameworks internal design specifications such that each chapter is dedicated for a certain framework. As explained later on, each framework's chapter will be divided into sections such that each section provides the internal design specifications for a certain component in that framework.

(2) $\langle CIDSD\_Chapter\_SharedPool_k \rangle_{k=0}^{s}$: this element represents the sequence of chapters that provide the shared pools internal design specifications such that each chapter is dedicated for a certain one of these shared pools. As explained later on, each

shared pool's chapter will be divided into sections such that each section provides the internal design specifications for a certain class in that shared pool.

$$CIDSD \equiv \left[ \begin{array}{l} \langle CIDSD\_Chapter\_Framework_i \rangle_{i=1}^{F}, \\ \langle CIDSD\_Chapter\_SharedPool_k \rangle_{k=0}^{S} \end{array} \right] \tag{6–12}$$

***CISD*s** in (6–3) above do not involve the shared pools, while the internal design specifications documents, ***CIDSD*s** does. The justification for this is that all shared entities are used internally by the components from the different frameworks and they are not intended to be directly used by externals.

### 6.5.2.3.1 Explanation of (6–12): *CDISD*

In formula (6–13), the internal design specifications of the $i^{th}$ framework (i.e. $CIDSD\_Chapter\_Framework_i$ of (6–12) above) is represented by a set of sections such that each section provides the internal design specifications for a certain component (i.e. $CIDSD\_Section\_Component_{ij}$).

$$CIDSD\_Chapter\_Framework_i \equiv \langle CIDSD\_Section\_Component_{ij} \rangle_{j=1}^{c_i} \tag{6–13}$$

In formula (6–14), the internal design specifications of the $k^{th}$ shared pool (i.e. $CIDSD\_Chapter\_SharedPool_k$ of (6–12) above) is represented by a tuple of the following two elements:

(1) $\langle CIDSD\_Section\_SharedClass_{kj} \rangle_{j=0}^{c^s_k}$: as explained in (6–16), this element represents the first sequence of the sections of the $k^{th}$ shared pool chapter. Each of these sections is dedicated for the internal design specifications of a certain class in that shared

pool. As explained in the lexicon of this view (*i.e. section 6.5.4*), $c_k^i$ is used to represent the number of shared classes in the $k^{th}$ shared pool.

(2) $\left\langle CIDSD\_Section\_OtherSharedEntity_{kj} \right\rangle_{j=0}^{e_k^i}$: this element represents the second sequence of sections of the $k^{th}$ shared pool chapter. Each of these sections is dedicated for the internal design specifications of a non–class entity in the $k^{th}$ shared pool. As explained in the lexicon of this view (*i.e. section 6.5.4*), $e_k^i$ is used to represent the number of non–class shared entities in the $k^{th}$ shared pool. It is left up to the developers to decide what non–class entities to be included in this chapter.

$$CIDSD\_Chapter\_SharedPool_k \equiv \begin{bmatrix} \left\langle CIDSD\_Section\_SharedClass_{kj} \right\rangle_{j=o}^{c_k^i}, \\ \left\langle CIDSD\_Section\_OtherSharedEntity_{kj} \right\rangle_{j=1}^{e_k^i} \end{bmatrix} \tag{6–14}$$

### 6.5.2.3.2 Explanation of (6–13): *CIDSD_Section_Component$_{ij}$*

In formula (6–15), the internal design specifications of the $ij^{th}$ component (i.e. *CIDSD_Section_Component$_{ij}$* of (6–13) above) is represented by a tuple of the following two elements.

(1) *CIDSD_SubSection_Main$_{ij}$*: as explained in (6–17), this element represents the internal design specifications of the main section of the $ij^{th}$ component. The main section of any component is mainly a straight code that enables users to use the given component as a complete application rather than just invoking some of its classes. Parnas DISPLAY method [40] can be used to document the internal design of components main sections.

(2) $\langle CIDSD\_SubSection\_InternalClass_{ijl} \rangle_{l=1}^{c_{ij}}$: as explained in (6–18), this element represents the sequence of sub-sections that provide the internal design specifications of the internal classes of the $ij^{th}$ component. Each of these sub-sections is dedicated for one of these internal classes. The $c_{ij}$ is used to represent the number of internal classes in the $ij^{th}$ component. As explained later on, each internal class is composed of up to three parts: public, protected, and private. The public part can contain only public methods (access programs) that can be invoked by external users. The protected part has mainly protected methods, but still it may contain some protected objects. The private part has mainly private objects (i.e. data structure). The internal design of each of these methods whether public, protected, or even private can be specified "documented" using Parnas DISPLAY method [40] similar to the main section explained above.

$$CIDSD\_Section\_Component_{ij} \equiv \begin{bmatrix} CIDSD\_SubSection\_Main_{ij}, \\ \langle CIDSD\_SubSection\_InternalClass_{ijl} \rangle_{l=1}^{c_{ij}} \end{bmatrix} \qquad (6\text{–}15)$$

We should notice that the internal design specification of each component are divided into specifications for the main section and specifications for the internal classes of that component. This is consistent with how we viewed the implementation of components in the previous chapter's design and implementation view.

#### 6.5.2.3.3 Explanation of (6–14): $CIDSD\_Chapter\_SharedPool_k$

In formula (6–16), the internal design specifications of the $kj^{th}$ shared class (i.e. $CIDSD\_Section\_SharedClass_{kj}$ of (6–14) above) is represented by a tuple of the following three elements:

(1) $\mathit{CIDSD\_SubSection\_SharedClass\_Public}_{kj}$: as explained in (6–19), this element represents the sub–section that provides the internal design specifications of the public section of the $kj^{th}$ shared class.

(2) $\mathit{CIDSD\_SubSection\_SharedClass\_Protected}_{kj}$: as explained in (6–20), this element represents the sub–section that provides the internal design specifications of the protected section of the $kj^{th}$ shared class.

(3) $\mathit{CIDSD\_SubSection\_SharedClass\_Private}_{kj}$: as explained in (6–21), this element represents the sub–section that provides the internal design specifications of the private section of the $kj^{th}$ shared class.

$$\mathit{CIDSD\_Section\_SharedClass}_{kj} \equiv \begin{bmatrix} \mathit{CIDSD\_SubSection\_SharedClass\_Public}_{kj}, \\ \mathit{CIDSD\_SubSection\_SharedClass\_Protected}_{kj}, \\ \mathit{CIDSD\_SubSection\_SharedClass\_Private}_{kj} \end{bmatrix} \tag{6–16}$$

### 6.5.2.3.4 Explanation of (6–15): $\mathit{CIDSD\_Section\_Component}_{ij}$

In formula (6–17), we show that Parnas DISPLAY method [39] can be used to specify the internal design of each component's main part.

$$\mathit{CIDSD\_SubSection\_Main}_{ij} \equiv \left\langle \mathit{CIDSD\_SubSection\_Main\_DISPLAY}_{ij} \right\rangle_{d=1}^{\mathit{NumberOfDISPLAYsForMainPartOfComponent}_{ijd}} \tag{6–17}$$

In formula (6–18), the internal design specifications of the $l^{th}$ internal class of the $ij^{th}$ component (i.e. $\mathit{CIDSD\_SubSection\_InternalClass}_{ijl}$ of (6–15) above) is represented by a tuple of the following three elements:

(1) $\mathit{CIDSD\_SubSection\_InternalClass\_Public}_{ijl}$: as explained in (6–22), this element represents the sub–section that provides the internal design specifications of the public section of the $l^{th}$ internal class of the $ij^{th}$ component.

(2) $\textit{CIDSD\_SubSection\_InternalClass\_Protected}_{ijl}$: as explained in (6–23), this element represents the sub–section that provides the internal design specifications of the protected section of the $l^{th}$ internal class of the $ij^{th}$ component.

(3) $\textit{CIDSD\_SubSection\_InternalClass\_Private}_{ijl}$: as explained in (6–24), this element represents the sub–section that provides the internal design specifications of the private section of the $l^{th}$ internal class of the $ij^{th}$ component.

$$\textit{CIDSD\_SubSection\_InternalClass}_{ijl} \equiv \begin{bmatrix} \textit{CIDSD\_SubSection\_InternalClassPublic}_{ijl}, \\ \textit{CIDSD\_SubSection\_InternalClassProtected}_{ijl}, \\ \textit{CIDSD\_SubSection\_InternalClassPrivate}_{ijl} \end{bmatrix} \quad (6\text{–}18)$$

#### 6.5.2.3.5 Explanation of (6–16): $\textit{CIDSD\_Section\_SharedClass}_{ij}$

Parnas DISPLAY method can be used to specify the internal design of shared classes.

$$\textit{CIDSD\_SubSection\_SharedClass\_Public}_{ij} \equiv \text{this is similar to (6–22), i.e. public parts} \quad (6\text{–}19)$$

$$\textit{CIDSD\_SubSection\_SharedClass\_Protected}_{ij} \equiv \text{this is similar to (6–23), i.e. protected parts} \quad (6\text{–}20)$$

$$\textit{CIDSD\_SubSection\_SharedClass\_Private}_{ij} \equiv \text{this is similar to (6–24), i.e. private parts.} \quad (6\text{–}21)$$

#### 6.5.2.3.6 Explanation of (6–18): $\textit{CIDSD\_SubSection\_InternalClass}_{ij}$

In formula (6–22), the internal design specifications for the public parts of internal classes can be done using Parnas DISPLAYs method. The internal design specifications for each internal class's public part is represented by $\bigcup_{m=1}^{M^P_{ijl}} \left\{ \left\langle \textit{InternalClass\_PublicMethod\_DISPLAY}_{ijlmd} \right\rangle_{d=1}^{\textit{NumberOfDISPLAYsForPublicMethod}_{ijlm}} \right\}$. This element represents a set of sequences of DISPLAYs. Each of these sequences (i.e. $\left\langle \textit{InternalClass\_PublicMethod\_DISPLAY}_{ijlmd} \right\rangle_{d=1}^{\textit{NumberOfDISPLAYsForPublicMethod}_{ijlm}}$) represents the subset of Parnas DISPLAYs used to specify the internal design of a certain public method. As per the the lexicon of this view (*section 6.5.4*), $M^P_{ijl}$. represents the number of public methods in the $l^{th}$ class of the $ij^{th}$ component. $NumberOfDISPLAYsForPublicMethod_{ijlm}$ represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ public method in the $l^{th}$ class of the $ij^{th}$ component.

$$CIDSD\_SubSection\_InternalClass\_Public_{ijl} \equiv \bigcup_{m=1}^{M^{P}_{ijl}} \left\{ \langle InternalClass\_PublicMethod\_DISPLAY_{ijlmd} \rangle_{d=1}^{NumberOfDISPLAYsForPublicMethod_{ijlm}} \right\} \quad (6\text{–}22)$$

In formula (6–23), the internal design specifications for the protected parts of internal classes can be done using Parnas DISPLAYs method similar to the public part above.

$$CIDSD\_SubSection\_InternalClass\_Protected_{ijl} \equiv \bigcup_{m=0}^{M^{R}_{ijl}} \left\{ \langle InternalClass\_ProtectedMethod\_DISPLAY_{ijlmd} \rangle_{d=1}^{NumberOfDISPLAYsForProtectedMethod_{ijlm}} \right\} \quad (6\text{–}23)$$

In formula (6–24), the internal design specifications for the private parts of internal classes can be done using Parnas DISPLAYs method similar to the public and protected parts above. The internal design specifications for each internal class's private part is represented by a tuple of the following two elements:

(1) $\bigcup_{m=0}^{M^{V}_{ijl}} \left\{ \langle InternalClass\_PrivateMethod\_DISPLAY_{ijlmd} \rangle_{d=1}^{NumberOfDISPLAYsForPrivateMethod_{ijlm}} \right\}$: similar to the above, this element represents a set of sequences of DISPLAYs. Each of these sequences (i.e. $\langle InternalClass\_PrivateMethod\_DISPLAY_{ijlmd} \rangle_{d=1}^{NumberOfDISPLAYsForPrivateMethod_{ijlm}}$) represents the subset of Parnas DISPLAYs used to specify the internal design of a certain private method. As per the the lexicon of this view (*section 6.5.4*), $M^{V}_{ijl}$. represents the number of private methods in the $l^{th}$ class of the $ij^{th}$ component. $NumberOfDISPLAYsForPrivateMethod_{ijlm}$ rep-

resents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ private method in the $l^{th}$ class of the $ij^{th}$ component.

(2) $InternalClass\_DataStructureSpecifications_{ijl}$: this element describes the data structure implemented in the $l^{th}$ class of the $ij^{th}$ component. Readers are reminded that a component's data structure is the composition of the data structures of that component's internal classes.

$$CIDSD\_SubSection\_InternalClass\_Private_{ijl} \equiv \begin{bmatrix} \bigcup_{m=0}^{M_{ijl}^{V}} \left\{ \left\langle InternalClass\_PrivateMethod\_DISPLAY_{ijlmd} \right\rangle_{d=1}^{NumberOfDISPLAYsForPrivateMethod^{ijlm}} \right\}, \\ InternalClass\_DataStructureSpecifications_{ijl} \end{bmatrix} \quad (6\text{–}24)$$

### 6.5.3 The View Characteristics

The following are the main points that characterize this third view:

(1) A CO–Toolkit's CGD document is an informal document. Therefore, it may contain various information and details about that CO–Toolkit. CO–Toolkits' Developer are the ones who should decide what to contain in their CGDs.

(2) A CO–Toolkit's CISD document must address each component in that CO–Toolkit. This is because of that a CO–Toolkit's CISD reflect how each component should be used.

(3) A CO–Toolkit's CISD should not address any internal classes of components as well as should not address any shared class from any shared pool. This is because of that a CO–Toolkit's CISD is intended for providing interface specifications and should not reveal and internal specifications.

(4) A CO–Toolkit's CIDSD must address the main section and the internal classes of each component in that CO–Toolkit. Also, it must address the shared classes from the different shared pools. This is because of that a CO–Toolkit's CIDSD reflects

the internal design of each component as well as the internal design of shared pools' classes and entities.

(5) The various documents (i.e. CGD, CISD, CIDSD, etc.) have to be respect the characteristics of the other views that we introduced in the previous two chapters. This is because of that proper documents always reflect their systems' design, implementation, and interface specifications.

## 6.5.4 The View Lexicon

### 6.5.4.1 Variables And Acronyms

*COToolkit*:Represents any Component–Oriented Toolkit that can provide reusable components to be used to build new applications.

*CO*: Component Oriented.

#### 6.5.4.1.1 CGD's Variables And Acronyms

*CGD*: Represents the **Component Guide Document** as described in section 6.2.

*CGD_DescriptiveOverview*: Represents the first chapter of the proposed CGD document. This chapter should describe the various aspects of the CO–Toolkit under consideration.

*CGD_PictorialOverview*: Represents the second chapter of the proposed CGD document. This chapter should provide a set of pictorial views that describe several things such as the file structure used for the considered CO–Toolkit, the various relations across the frameworks and among their components (i.e. such as inheritance and association relations), and etc.

*CGD_IndexedQuickReferenceTables*: Represents the third chapter of the proposed CGD document. This chapter should provide a set of index table that can help users as well as maintainers to find out what service(s) is (are) provided by each component in the CO–Toolkit under consideration.

#### 6.5.4.1.2 CISD's Variables And Acronyms

*CISD*: Represents the **Component Interface Specifications Document** as described in section 6.3.

*CISD_Chapter_Framework*$_i$: Represents the CISD chapter that provides the interface specifications for the $i^{th}$ framework.

*CISD_Section_Component*$_{ij}$: Represents the section dedicated for the interface specifications of $ij^{th}$ component as a part of the CISD chapter for the $i^{th}$ framework.

***CISD_SubSection_CHARACTERISTICS***$_{ij}$: Represents the general characteristics of the $ij^{th}$ component.

***CISD_SubSection_SYNTAX***$_{ij}$: Represents the signatures of the access programs of the $ij^{th}$ component.

***CISD_SubSection_CANONICAL_REPRESENTATION***$_{ij}$: Represents the canonical representation for the $ij^{th}$ component.

***CISD_SubSection_EQUIVALENCES***$_{ij}$: Represents the equivalences for the $ij^{th}$ component.

***CISD_SubSection_VALUES***$_{ij}$: Represents the values returned by the access programs of the $ij^{th}$ component.

***CISD_SubSection_LEXICON***$_{ij}$: Represents the lexicon for the interface specifications of the $ij^{th}$ component.

***TypeSpecified***$_{ij}$: Represents the specification type of the $ij^{th}$ component.

***PublicPrimitiveSystemDataTypes***$_{ij}$: Represents the set of primitive data types (i.e. int, boolean etc.) that appear in the interface specifications of the $ij^{th}$ component.

***PublicConstantParameters***$_{ij}$: Represents the set of public constant that appear in the interface specifications of the $ij^{th}$ component.

***PublicSpecificationTypes***$_{ij}$: Represents the set of types that appear in the interface specifications of the $ij^{th}$ component.

***PublicSpecificationVariables***$_{ij}$: Represents the set of variables that appear in the interface specifications of the $ij^{th}$ component.

***Public_ACCESS_ProgramsInComponent***$_{ij}$: Represents the set of public access programs (methods) that are contained in the $ij^{th}$ component.

***InheritedPublic_ACCESS_ProgramsByComponent***$_{ij}$: Represents the set of access programs (methods) that are publicly inherited by the $ij^{th}$ component from other components.

***CANONICAL_REPRESENTATION_OfComponent***$_{ij}$: Represents the canonical representation for the $ij^{th}$ component.

***ReferencesTo_CANONICAL_REPRESENTATIONS_OFInheritedComponents***$_{ij}$: Represents a set of references to the canonical representations of the other components that are inherited from by the $ij^{th}$ component.

***CANONICAL_REPRESENTATION_OfInternalClass***$_{ijl}$: Represents the canonical representation for the $l^{th}$ internal class of the $ij^{th}$ component.

***EQUIVALENCES_OfComponent***$_{ij}$: Represents the set of equivalent states to the canonical one. An object of the $ij^{th}$ component goes into certain one of these equivalent states whenever a certain access program is invoked on that object.

***ReferencesTo_EQUIVALENCES_OFInheritedComponents$_{ij}$***: Represents a set of references to the equivalences of the other components that are inherited from by the $ij^{th}$ component.

***VALUES_OfComponent$_{ij}$***: Represents the set of values that are returned by the public access programs (methods) of the $ij^{th}$ component.

***ReferencesTo_VALUES_OFInheritedComponents$_{ij}$***: Represents a set of references to the values returned by the public access programs (methods) of the other components that are inherited from by the $ij^{th}$ component.

### 6.5.4.1.3 CIDSD Variables And Acronyms

*CIDSD*: Represents the Component Internal Design Specifications Document as described in section 6.4.

***CIDSD_Chapter_Framework$_i$***: Represents the CIDSD chapter that provides the internal design specifications for the $i^{th}$ framework.

***CIDSD_Chapter_SharedPool$_k$***: Represents the CIDSD chapter that provides the internal design specifications for the $k^{th}$ shared pool.

***CIDSD_Section_Component$_{ij}$***: Represents the section dedicated for the internal design specifications of $ij^{th}$ component as a part of the CIDSD chapter for the $i^{th}$ framework.

***CIDSD_Section_SharedClass$_{kj}$***: Represents the section dedicated for the internal design specifications of the $kj^{th}$ shared class as a part of the CIDSD chapter for the $k^{th}$ shared–pool.

***CIDSD_Section_OtherSharedEntity$_{kj}$***: Represents the section dedicated for the internal design of the $kj^{th}$shared non–class entities as a part of the CIDSD chapter for the $k^{th}$ shared–pool.

***CIDSD_SubSection_Main$_{ij}$***: Represents the sub–section that provides the internal design specifications of the main part of the $ij^{th}$ component.

***CIDSD_SubSection_InternalClass$_{ij}$***: Represents the sub–section that provides the internal design specifications for the $l^{th}$ internal class in the $ij^{th}$ component.

***CIDSD_SubSection_SharedClass_Public$_{kj}$***: Represents the sub–section that provides the internal design specifications of the public section of the $kj^{th}$ shared class.

***CIDSD_SubSection_SharedClass_Protected$_{kj}$***: Represents the sub–section that provides the internal design specifications of the protected section of the $kj^{th}$ shared class.

***CIDSD_SubSection_SharedClass_Private$_{kj}$***: as explained in (6–21), this element represents the internal design specifications of the private section of the $kj^{th}$ shared class.

***CIDSD_SubSection_Main_DISPLAY$_{ijd}$***: Represents the $d^{th}$ DISPLAY of those used to specify the internal design of the Main part of the $ij^{th}$ component.

***CIDSD_SubSection_InternalClass_Public$_{ijl}$***: Represents the sub–section that provides the internal design specifications of the public section of the $l^{th}$ internal class of the $ij^{th}$ component.

$CIDSD\_SubSection\_InternalClass\_Protected_{ijl}$: Represents the sub-section that provides the internal design specifications of the protected section of the $l^{th}$ internal class of the $ij^{th}$ component.

$CIDSD\_SubSection\_InternalClass\_Private_{ijl}$: Represents the sub-section that provides the internal design specifications of the private section of the $l^{th}$ internal class of the $ij^{th}$ component.

$InternalClass\_PublicMethod\_DISPLAY_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ public method in the $l^{th}$ internal class of the $ij^{th}$ component.

$ReferencesTo\_DISPLAYs\_OFPublicMethodsInheritedByInternalClass_{ijl}$: Represents a set of references to the set of Parnas DISPLAYs that provide the internal specifications for those internal class methods that are inherited by the $l^{th}$ internal class of the $ij^{th}$ component.

$InternalClass\_ProtectedMethod\_DISPLAY_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ protected method in the $l^{th}$ internal class of the $ij^{th}$ component.

$InternalClass\_PrivateMethod\_DISPLAY_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ private method in the $l^{th}$ internal class of the $ij^{th}$ component.

$bldInternalClass\_DataStructureSpecifications_{ijl}$: Represents the data structure implemented in the $l^{th}$ class of the $ij^{th}$ component.

### 6.5.4.2 Symbols And Constants

$F$: Represents the number of frameworks in the given CO_toolkit.

$S$: Represents the number of shared-pools.

$C_i$: Represents the number of components in the $i^{th}$ framework.

$c_k^s$: Represents the number of shared classes in the $k^{th}$ shared-pool.

$e_k^s$: Represents the number of other shared entities in the $k^{th}$ shared-pool.

$c_{ij}$: Represents the number of local classes used to construct the $ij^{th}$ component.

$M_{ijl}^P$: Represents the number of local public methods in the $l^{th}$ class of the $ij^{th}$ component.

$M_{ijl}^R$: Represents the number of local protected methods in the $l^{th}$ class of the $ij^{th}$ component.

$M_{ijl}^V$: Represents the number of local private methods in the $l^{th}$ class of the $ij^{th}$ component.

$NumberOfDISPLAYsForMainPartOfComponent_{ij}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the main part of the $ij^{th}$ component.

$NumberOfDISPLAYsForPublicMethod_{ijlm}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ public method in the $l^{th}$ internal class of the $ij^{th}$ component.

$NumberOfDISPLAYsForProtectedMethod_{ijlm}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ protected method in the $l^{th}$ internal class of the $ij^{th}$ component.

***NumberOfDISPLAYsForPrivateMethod***$_{ijlm}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ private method in the $l^{th}$ internal class of the $ij^{th}$ component.

## 6.6 Conclusion

In this chapter, we have formalized how to use "table based documentation methods" [7, 39, 40] to document CO-Toolkits.

# CHAPTER 7

# INITIAL CONDITIONS THAT MAKE THE THREE VIEWS CONSISTENT

## 7.1 Overview

CO–Toolkits have been viewed in this thesis from three separate perspectives: (i) their file structures (*c.f. chapter 4*), (ii) their design and implementation (*c.f. chapter 5*), and (iii) their documentation (*c.f. chapter 6*). The rules in these three views do not guarantee that the three views are describing the same system. However, they guarantee that it is not impossible that the three views are describing the same system. Satisfying the conditions that are presented in this chapter can improve the possibility of having the three views to be describing the same system. Some of these conditions relate all three views, while some others relate only pairs of views.

This chapter has six sections. The next section presents the conditions that should be satisfied by the three views together. The third section presents the conditions that should be satisfied by the file structure view (i.e. the view presented in chapter 4), and the design and implementation view (i.e. the view presented in chapter 5). The fourth section presents the conditions that should be satisfied by the design and implementation view, and the documentation view (i.e. the view presented in chapter 6). The fifth section presents the conditions that should be satisfied by the file structure view, and the documentation view. The last section describes this chapter's conclusions. All of these conditions should be satisfied in order to make the three views consistent.

## 7.2 Conditions That Relate All Three Views

This section presents the conditions that should be satisfied by the three views at the same time. The following two auxiliary functions are used for convenience in this section in order to maintain simple formulas. Also, these two functions are used in section 7.4.

*CARD(FLAG,X)* ≡ This is an auxiliary function that returns the number of elements in the set X. For convenience and as shown in the next table, the *FLAG* is used to indicate the type of elements in the set X.

| *FLAG* | **Type of Elements in Set X** |
|---|---|
| 1 | Each element in X is a "Set of Files". |
| 2 | Each element in X is a "Framework". |
| 3 | Each element in X is a "Component". |
| 4 | Each element in X is a "Shared Pool". |
| 5 | Each element in X is a "Class" |

*LENGTH(FLAG,Y)* ≡This function returns the number of elements in the sequence Y. For convenience and as shown in the next table, the FLAG is used to indicate the type of elements in the sequence Y.

| *FLAG* | **Type of Elements in Sequence Y** |
|---|---|
| 1 | Each element in Y is a "Chapter". |
| 2 | Each element in Y is a "Section". |
| 3 | Each element in Y is a "SubSection". |

(1) Concerning the following formulas:

** Formula (4–1) $CO\_Files \equiv \bigcup_{i=1}^{F} Fr\_Files_i \cup \bigcup_{k=0}^{S} Sh\_Files_k$ from the first view,

** Formula (5–1) $COToolkit \equiv \bigcup_{i=1}^{F} Framework_i \bigcup \bigcup_{k=0}^{S} SharedPool_k$ from the second view,

** Formula (6–3) $CISD \equiv \langle CISD\_Chapter\_Framework_i \rangle_{i=1}^{F}$ from the third view's CISD, and

** Formula (6–12) $CIDSD \equiv \left[\begin{array}{c} \langle CIDSD\_Chapter\_Framework_i \rangle_{i=1}^{F}, \\ \langle CIDSD\_Chapter\_SharedPool_k \rangle_{k=0}^{S} \end{array}\right]$ from the third view's CIDSD, the following conditions should be satisfied:

*i–* $F$ (i.e. the number of frameworks) should have the same value in the three views. More precisely, the predicate $(P1 \wedge P2 \wedge P3)$ should be TRUE, where $P1$, $P2$, and $P3$ are defined next. The justification for requiring this condition is simple. If the three views do not have the same number of frameworks, then either they do not describe the same system, or they fail their attempt to provide a correct description for the the same system.

*– $P1 \equiv \left( CARD\left(1, \bigcup_{i=1}^{F} Fr\_Files_i\right) = CARD\left(2, \bigcup_{i=1}^{F} Framework_i\right)\right)$: This predicate is to ensure that in the number of Frameworks' sets of files in the view for CO–Toolkits' File Structures (*i.e. the first view that is presented in chapter 4*) is the same as the number of Frameworks in the view for CO–Toolkits' Design and Implementation (*i.e. the second view that is presented in chapter 5*).

*– $P2 \equiv \left( CARD\left(1, \bigcup_{i=1}^{F} Fr\_Files_i\right) = LENGTH\left(1, \langle CISD\_Chapter\_Framework_i \rangle_{i=1}^{F}\right)\right)$: This predicate is to ensure that the number of Frameworks' sets of files in the first view is the same as the number of Frameworks' Chapters in the CISD document of the view for CO–Toolkits' Documentation (*i.e. the third view that is presented in chapter 6*).

*– $P3 \equiv \left( CARD\left(1, \bigcup_{i=1}^{F} Fr\_Files_i\right) = LENGTH\left(1, \langle CIDSD\_Chapter\_Framework_i \rangle_{i=1}^{F}\right)\right)$: This predicate is to ensure that the number of Frameworks' sets of files in the first view is the same as the number of Frameworks' Chapters in the CIDSD document of the third view.

*ii–* $S$ (i.e. the number of shared pools) should have the same value in the three views. More precisely, the predicate $(PS1 \wedge PS2)$ should be TRUE, where $PS1$, and $PS2$ are defined next. The justification for requiring this condition follows from the justification made above for $F$.

*– $PS1 \equiv \left( CARD\left(1, \bigcup_{k=0}^{S} Sh\_Files_k\right) = CARD\left(4, \bigcup_{k=0}^{S} SharedPool_k\right)\right)$: This predicate is to ensure that the number of Shared Pools' sets of files in the first view (*c.f. chapter 4*) is the same as the number of Shared Pools in the second view (*c.f. chapter 5*).

*– $PS2 \equiv \left( CARD\left(1, \bigcup_{k=0}^{S} Sh\_Files_k\right) = LENGTH\left(1, \langle CIDSD\_Chapter\_SharedPool_k \rangle_{k=0}^{S}\right)\right)$:

This predicate is to ensure that the number of Shared Pools' sets of files in the first view is the same as the number of Shared Pools' Chapters in the CIDSD document of the third view (c.f. chapter 6).

(2) Concerning the following formulas:

** Formula (4–2) $Fr\_Files_i \equiv \bigcup_{j=1}^{C_i} C\_Files_{ij}$ from the first view,

** Formula (5–2) $Framework_i \equiv \bigcup_{j=1}^{C_i} Component_{ij}$ from the second view,

** Formula (6–4) $CISD\_Chapter\_Framework_i \equiv \langle CISD\_Section\_Component_{ij} \rangle_{j=1}^{C_i}$ from the third view's CISD, and

** Formula (6–13) $CIDSD\_Chapter\_Framework_i \equiv \langle CIDSD\_Section\_Component_{ij} \rangle_{j=1}^{C_i}$ from the third view's CIDSD, the following condition should be satisfied:

*i–* For all $i$, $C_i$ that represent the number of components in the $i^{th}$ framework, should have the same value in the three views. More precisely, the predicate $(\forall i, (1 \le i \le F) \Rightarrow (PC1 \wedge PC2 \wedge PC3))$ should be TRUE, where $PC1$, $PC2$, and

*PC3* are defined next. The justification for requiring this condition follows from the justification above for requiring to have the same value for ***F*** in the three views.

- $PC1 \equiv \left( CARD\left(1, \bigcup_{j=1}^{C_i} C\_Files_{ij}\right) = CARD\left(3, \bigcup_{j=1}^{C_i} Component_{ij}\right)\right)$: This predicate is to ensure that the number of Components' sets of files of the $i^{th}$ Framework in the first view (*c.f. chapter 4*) is the same as the number of Components in the $i^{th}$ Framework in the second view (*c.f. chapter 5*)

- $PC2 \equiv \left( CARD\left(1, \bigcup_{j=1}^{C_i} C\_Files_{ij}\right) = LENGTH\left(2, \langle CISD\_Section\_Component_{ij} \rangle_{j=1}^{C_i}\right)\right)$:

  This predicate is to ensure that the number of Components' sets of files of the $i^{th}$ Framework in the first view is the same as the number of Components' Sections in the $i^{th}$ Framework's Chapter in the CISD document of the third view (*c.f. chapter 6*)

- $PC3 \equiv \left( CARD\left(1, \bigcup_{j=1}^{C_i} C\_Files_{ij}\right) = LENGTH\left(2, \langle CIDSD\_Section\_Component_{ij} \rangle_{j=1}^{C_i}\right)\right)$:

  This predicate is to ensure that the number of Components' sets of files of the $i^{th}$ Framework in the first view is the same as the number of Components' Sections in the $i^{th}$ Framework's Chapter in the CIDSD document of the third view (*c.f. chapter 6*)

## 7.3 Conditions That Relate The First Two Views

This section presents the conditions (i.e. in addition to those in section 7.2) that should be satisfied at the same time by the first two views (*c.f. chapters 4, and 5*). These initial conditions are required in order to make these two views consistent. The following auxiliary predicate is used for convenience in order to simplify the formulas that are presented in this section.

*UsedToImplement(X, Y)* $\equiv$ This is an auxiliary predicate that returns TRUE if the Source Code of the Sub-System Y is provided in the Set of Files X.

(1) Concerning formula (4–1) (i.e. $CO\_Files \equiv \bigcup_{i=1}^{F} Fr\_Files_i \bigcup \bigcup_{k=0}^{S} Sh\_Files_k$) from the first view, and formula (5–1) (i.e.$COToolkit \equiv \bigcup_{i=1}^{F} Framework_i \bigcup \bigcup_{k=0}^{S} SharedPool_k$) from the second view, the following conditions should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow (UsedToImplement(Fr\_Files_i, Framework_i)$=TRUE): This predicate is to ensure that each Framework has been implemented in its own set of files.

*ii–* $\forall k, (1 \le i \le S) \Rightarrow (UsedToImplement(Sh\_Files_k, SharedPool_k)$=TRUE): This predicate is to ensure that each Share Pool has been implemented in its own set of files.

(2) Concerning formula (4–2) (i.e. $Fr\_Files_i \equiv \bigcup_{j=1}^{C_i} C\_Files_{ij}$) from the first view, and formula (5–2) (i.e. $Framework_i \equiv \bigcup_{j=1}^{C_i} Component_{ij}$) from the second view, the following condition should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow (UsedToImplement(C\_Files_{ij}, Component_{ij})$=TRUE): This predicate is to ensure that each component has been implemented in its own set of files.

(3) Concerning formula (4–4) (i.e. $C\_Files_{ij} \equiv \{topFile_{ij}, intermediateFile_{ij}, internalFile_{ij}\}$) from the first view, and formula (5–4) $Component_{ij} \equiv \begin{bmatrix} Interface_{ij}, \\ Implementation_{ij} \end{bmatrix}$ and its sub–formulas (5–5) through (5–17) from the second view, the following conditions should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow (UsedToImplement(topFile_{ij}, MainPart_{ij})$=TRUE): This predicate is to ensure that the implementation of the main part of each component is provided in the top file of that component. The term $MainPart_{ij}$ is addressed in the sub–formula (5–10) of chapter 5.

*ii–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow (UsedToImplement(IntermediateFile_{ij}, Interface_{ij})$=TRUE): This predicate is to ensure that the interface of each component is provided in the intermediate file of that component.

*iii–* $\forall i, (1 \le i \le F) \rightarrow \forall j, (1 \le j \le C_i) \rightarrow (\mathit{UsedToImplement}(\mathit{InternalFile}_{ij}, \mathit{Implementation}_{ij}) = \mathrm{TRUE})$

This predicate is to ensure that the implementation of each component is provided in the internal file of that component.

## 7.4 Conditions That Relate The Second And Third Views

This section presents the conditions (i.e. in addition to those in section 7.2) that should be satisfied at the same time by the second view (*c.f. chapter 5*) and third view (*c.f. chapter 6*). These initial conditions are required in order to make these two views consistent.

There are three documents considered in the third view (i.e. CGD, CISD, and CIDSD). We are concerned about the conditions that relate the second view and the third view's CISD and CIDSD documents since the CGD document is an informal one.

### 7.4.1 Conditions That Relate The Second View And Third View's CISD Document

This section presents the conditions (i.e. in addition to those in section 7.2) that should be satisfied at the same time by the second view (*c.f. chapter 5*), and the CISD document of the documentation view (*c.f. chapter 6: section 6.5.2.2*). The following auxiliary predicate is used for convenience throughout this sub–section in order to maintain simple formulas.

*ProvidesInterfaceSpecifications*$(X, Y)$ $\equiv$ This is an auxiliary predicate that returns TRUE if the interface specifications of the Sub–System Y are provided in the portion X of the CISD document.

(1) Concerning formula (5–1) (i.e. $\mathit{COToolkit} \equiv \bigcup_{i=1}^{F} \mathit{Framework}_i \bigcup \bigcup_{k=0}^{S} \mathit{SharedPool}_k$) from the second view, and formula (6–3) (i.e. $\mathit{CISD} \equiv \langle \mathit{CISD\_Chapter\_Framework}_i \rangle_{i=1}^{F}$) from the third view's CISD, the following condition should be satisfied:

i– $\forall i, (1 \le i \le F) \Rightarrow (ProvidesInterfaceSpecifications(CISD\_Chapter\_Framework_i, Framework_i) = \text{TRUE})$:

This predicate is to ensure that the interface specifications for each framework is provided in the CISD' Chapter that is dedicated for that framework.

(2) Concerning formula (5–2) (i.e. $Framework_i \equiv \bigcup_{j=1}^{C_i} Component_{ij}$) from the second view and the formula (6–4) (i.e. $CISD\_Chapter\_Framework_i \equiv \langle CISD\_Section\_Component_{ij} \rangle_{j=1}^{C_i}$) from the third view's CISD, the following condition should be satisfied:

i– $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow$

$$(ProvidesInterfaceSpecifications(CISD\_Section\_Component_{ij}, Component_{ij}) = \text{TRUE}):$$

This predicate is to ensure that the interface specification for each component is provided in the CISD' Section that is dedicated for that component.

### 7.4.2 Conditions That Relate The Second View And Third View's CIDSD Document

This section presents the conditions (i.e. in addition to those in section 7.2) that should be satisfied at the same time by the second view (*c.f. chapter 5*) and third view's CIDSD document (*c.f. chapter 6, section 6.5.2.3.1*). The following auxiliary predicate is used (i.e. in addition to those two functions in section 7.2) for convenience throughout this sub–section in order to maintain simple formulas.

$ProvidesDesignSpecifications(X, Y) \equiv$ This is an auxiliary predicate that returns TRUE if the design and implementation specifications of the Sub–System Y are provided in the portion X of the CIDSD document.

(1) Concerning the following formulas:

** Formula (5–10) $InternalClassesSet_{ij} \equiv \bigcup_{l=1}^{c_{ij}} InternalClass_{ijl}$ from the second view, and

** Formula (6–15) $CIDSD\_Section\_Component_{ij} \equiv \begin{bmatrix} CIDSD\_SubSection\_Main_{ij}, \\ \overset{c_{ij}}{\underset{l=1}{\langle}} CIDSD\_SubSection\_InternalClass_{ijl} \rangle \end{bmatrix}$ from the third view's CIDSD, the following condition should be satisfied:

i– $c_{ij}$ that represents the number of internal classes in the $ij^{th}$ component, should have the same value in the second view and the third view's CIDSD documents. More precisely, the next predicate should be TRUE. The justification for this is that the number of internal classes in each component should be the same as the number of CIDSD's Sub–Sections that are dedicated for that component.

$$\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow$$
$$\left[ CARD(6,\ InternalClassSet_{ij}) = LENGTH\left(3,\ \overset{c_{ij}}{\underset{l=1}{\langle}} CIDSD\_SubSection\_InternalClass_{ijl} \rangle \right)\right]$$

(2) Concerning the following formulas:

** Formula (5–3) $SharedPool_k \equiv \begin{bmatrix} SharedClassesSet_k, \\ OtherSharedEntitiesSet_k \end{bmatrix}$, where

$SharedClassesSet_k \equiv \bigcup_{j=0}^{c_k^s} SharedClass_{kj}$ , and $OtherSharedEntitiesSet_k \equiv \bigcup_{f=1}^{c_k^e} SharedEntity_{kf}$

from the second view, and

** Formula (6–14) $CIDSD\_Chapter\_SharedPool_k \equiv \begin{bmatrix} \overset{c_k^s}{\underset{j=0}{\langle}} CIDSD\_Section\_SharedClass_{kj} \rangle, \\ \overset{c_k^e}{\underset{j=1}{\langle}} CIDSD\_Section\_OtherSharedEntity_{kj} \rangle \end{bmatrix}$ from

the third view's CIDSD, the following condition should be satisfied:

i– $c_k^s$ that represents the number of shared classes in the $k^{th}$ shared–pool should have the same value in both the second view and the third view's CIDSD document. More precisely, the next predicate should be TRUE. The justification for this is that the number of shared classes in each Shared Pool should be the same as the number of CIDSD's

Sections that are dedicated for the shared classes of that Shared Pool.

$$\forall k, (0 \le k \le S) \Rightarrow$$
$$\left[CARD\left[5, \bigcup_{j=0}^{c_k^s} SharedClass_{kj}\right] = LENGTH\left[2, \left\langle CIDSD\_Section\_SharedClass_{kj}\right\rangle_{j=0}^{c_k^s}\right]\right]$$

(3) Concerning formula (5–1) (i.e. $COToolkit \equiv \bigcup_{i=1}^{F} Framework_i \bigcup \bigcup_{k=0}^{S} SharedPool_k$) from the second view, and formula (6–12) (i.e. $CIDSD \equiv \begin{bmatrix} \left\langle CIDSD\_Chapter\_Framework_i\right\rangle_{i=1}^{F}, \\ \left\langle CIDSD\_Chapter\_SharedPool_k\right\rangle_{k=0}^{S} \end{bmatrix}$) from the third view's CIDSD, the following conditions should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow$

$(ProvidesDesignSpecifications(CIDSD\_Chapter\_Framework_i, Framework_i) = \text{TRUE})$:

This predicate is to ensure that the design specifications for each framework is provided in the CIDSD' Chapter that is dedicated for that Framework.

*ii–* $\forall k, (1 \le i \le S) \Rightarrow$

$(ProvidesDesignSpecifications(CIDSD\_Chapter\_SharedPool_k, SharedPool_k) = \text{TRUE})$:

This predicate is to ensure that the design specifications for each Shared Pool is provided in the CIDSD' Chapter that is dedicated for that Shared Pool.

(4) Concerning formula (5–2) (i.e. $Framework_i \equiv \bigcup_{j=1}^{C_i} Component_{ij}$) from the second view, and formula (6–13) (i.e. $CIDSD\_Chapter\_Framework_i \equiv \left\langle CIDSD\_Section\_Component_{ij}\right\rangle_{j=1}^{C_i}$) from the third view's CIDSD, the following condition should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow$

$(ProvidesDesignSpecifications(CIDSD\_Section\_Component_{ij}, Component_{ij}) = \text{TRUE})$:

This predicate is to ensure that the design specifications for each component is provided in the CIDSD' Section that is dedicated for that component.

(5) Concerning formula (5–10) (i.e. $Implementation_{ij} \equiv \begin{bmatrix} MainPart_{ij}, \\ InternalClassesSet_{ij} \end{bmatrix}$ where $InternalClassesSet_{ij} \equiv \bigcup_{l=1}^{c_{ij}} InternalClass_{ijl}$) from the second view, and formula (6–15) (i.e. $CIDSD\_Section\_Component_{ij} \equiv \begin{bmatrix} CIDSD\_SubSection\_Main_{ij}, \\ \left\langle CIDSD\_SubSection\_InternalClass_{ijl} \right\rangle_{l=1}^{c_{ij}} \end{bmatrix}$) from the third view's CIDSD document, the following conditions should be satisfied:

*i–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow$

$(ProvidesDesignSpecifications(CIDSD\_SubSection\_Main_{ij}, MainPart_{ij})$=TRUE):

This predicate is to ensure that the design specifications for each component's main part is provided in the CIDSD' Sub–Section that is dedicated for that main part.

*ii–* $\forall i, (1 \le i \le F) \Rightarrow \forall j, (1 \le j \le C_i) \Rightarrow$

$(ProvidesDesignSpecifications(CIDSD\_SubSection\_InternalClass_{ijl}, InternalClass_{ijl})$=TRUE):

This predicate is to ensure that the design specifications for each internal class is provided in the CIDSD' Sub–Section that is dedicated for that internal class.

## 7.5 Conditions That Relate The First And Third Views

The conditions that need to be satisfied in order to ensure the consistency of the file structure view (i.e. the first view presented in chapter 4), and the documentation view (i.e. the third view presented in chapter 6) are:

(1) Satisfying the conditions that relate the three views (*c.f. section 7.2*).

(2) Satisfying the conditions that relate the first two views (*c.f. see section 7.3*).

(3) Satisfying the conditions that relate the second and the third views (*c.f. sections 7.4, 7.4.1, and 7.4.2*).

## 7.6 Conclusions

In this chapter, we have presented the conditions that should be satisfied by either the three views or by only pairs of these views at the same time. These conditions are required in order to make these three views consistent.

(1) Section 7.2 presents the conditions that should be satisfied by the three views together.

(2) Section 7.3 presents the conditions that should be satisfied by the file structure view and the implementation view.

(3) Section 7.4 presents the conditions that should be satisfied by the implementation view and the documentation view.

(4) Section 7.5 presents the conditions that should be satisfied by the file structure view and the documentation view.

# CHAPTER 8

# LIMITATIONS, CONCLUSIONS, AND FUTURE WORK

## 8.1 Overview

This chapter has four sections. The next section describes the limitations of our work. The third section describes the main conclusions about this research. The last section describes future work that can potentially improve and extend the outcome of this research.

## 8.2 Limitations Of This Research

The limitations of this research can be summarized as follows:

(1) In the view for for CO–Toolkits' file structures (*c.f. chapter 4*), we limited the file hierarchy within each framework to only three levels: top, intermediate, and internal. Although, this can be seen as a restriction, we think that this restriction makes it easier for us to formalize the set of rules for file structures that we derived based on the assumptions for good design practice (*c.f. sections 3.3 and 3.3.2*).

(2) Given that our prototype systems were intended to just demonstrate that formalized rules can be used to build tools, we did not provide any front ends for these prototypes just to keep them simple. Also, we think that existing front ends such DATRIX from Bell Canada can be customized and integrated with our prototypes.

## 8.3 Conclusions

By this research, we have shown the following:

(1) Rules for organizing, designing, and implementing CO–Toolkits can be derived based on a set of initial assumptions for good design practice *(c.f. chapter 3).*

(2) These rules can be formalized using an appropriate mathematical terminology and notation (*c.f. chapters 4, 5, and 6*).

(3) These formalized rules can be used to develop tools (*c.f. Appendixes A, and B*).

(4) Such tools can be used to check CO–Toolkits' code against these formalized rules and report any violations back to developers (*c.f. Appendixes A2–A4, and B2–B4*).

(5) Our views are intended to be general and language–independent. However, the following things need to be considered when using our views:

*i–* The view for file structures is good for languages that support the #include directive such as C, C++. For languages that do not support such directive (i.e. JAVA, FORTRAN, etc.), some of the rules needs to be modified to fit the equivalent directive (i.e. if there is any).

*ii*– The view for implementation structures is good for object–oriented languages. For example, it can not be used to check code that is written in C.

*iii*– Given that the documentation view is focusing on the interface specifications and the internal design specifications, then it can be used to document code that is written in any language.

(6) Experience shows that there is no other place that tried to make formal statement of rules for good design.

(7) Given that we are looking at a large set of badly defined artifacts and trying to find general rules, there may be no simple general rules in our field –unlike nature– because languages are often poorly designed.

## 8.4 Future Work

This section describes future works that can potentially extend this research:

(1) As our formal models are not complete, further work can be conducted in the future to extend and improve the set of rules presented in this thesis.

(2) Investigating the possibility of simplifying the mathematical notation and terminology that we used to formalize our rules described in chapters 4 through 6.

(3) Investigating the integration of existing front ends such as the DATRIX tool from Bell Canada into our prototypes.

(4) Investigating the use of design patterns in documenting CO–Toolkits. According to Fayad and Schmidt [19], "*Patterns represent recurring solutions to software development problems within a particular context*".

(5) In [28], Lakos provided a set of arbitrary and language–dependent rules that address components construction. Investigating how to formalize these rules can be performed in further research.

# APPENDIX A

# A PROTOTYPE FOR CO–TOOLKITS' FILE STRUCTURES

# APPENDIX A: CO–TOOLKITS' FILE STRUCTURES PROTOTYPE

# Appendix A1

# A Prototype For CO–Toolkits's File Structures: An Overview

## A1.1 Prototype Description

Java has been used to build this prototype based on the view for CO–Toolkits' file structures (*c.f. chapter 4*). The prototype's data structure represents the view structure of section 4.2.2. Each predicate from the view's characteristics section (*c.f. section 4.2.3*) is implemented as a method that returns a boolean value (i.e. TRUE or FALSE). For convenience, we named each of these methods by its formula number as in chapter 4.

Given that the purpose of building this prototype is to show that this view can be used to build a tool, we kept this prototype simple. All of its inputs are built–in. Currently, three built–in examples are provided, one of them is on ACE file structure (*c.f. Appendixes A2–A4*).

To run this prototype:

(1) The twelve files listed in the next section should be in a directory named **V1_FileStructure_R3**. In Java, path and file names are case sensitive.

(2) Run the command **javac –deprecation V1_FileStructure_R3/*.java**

(3) Run the command **java V1_FileStructure_R3.CheckFileStructure N.** N is the argument. It represents the example number of choice (i.e. 1 for example 1, 2 for example 2, and 3 for the ACE example).

The prototype operation goes through the following steps:

(1) It allocates its internal data structure based on the view structure of section 4.2.2. To perform this step, it requires the number of frameworks (i.e. ***F***), the number of shared pools (i.e. ***S***), the number of components in each framework (i.e. ***C[ ]***), and the number of shared files in each shared pool (***SC[ ]***).

(2) It updates its internal data structures with the file names and their extensions. For now, we use .top for top files, .inm for intermediate files, and .inl for internal files.

(3) It updates its internal data structure with the file "#include" relations.

(4) It checks its internal data structure by sequentially invoking its methods that represent the various predicate (i.e. rules) in the view characteristic section (*c.f. section 4.2.3*). Each of these methods reports any violation to its embedded rule.

# A1.2 Source Code

## A1.2.1 CheckFileStructure.java

This file contains the implementation of all predicates that represent the characteristics of the File Structure's View. These predicates are named by their formula numbers as in chapter 4. For example, predicate4–20 is the implementation of characteristic (4–20). Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class CheckFileStructure_R3 {
```

```
private static int Ptr=0;
private static boolean LoopOccured=false;
private static boolean MultiplePathOccured=false;
private static boolean MatrixExtracted=false;
private static int[][] InclusionMatrix= new int[((Globals.MAX_SP-1) * (Globals.MAX_SF -1)) + ((Globals.MAX_FR -1) * (Globals.MAX_CO -1) * (Globals.MAX_FL -1)) +1][((Globals.MAX_SP-1) * (Globals.MAX_SF -1)) + ((Globals.MAX_FR -1) * (Globals.MAX_CO -1) * (Globals.MAX_FL -1)) +1];
private static int M;          // for the InclusionMatrix dimension [MxM]
private static tmpFileNode[] ListFiles= new tmpFileNode[((Globals.MAX_FR-1)*(Globals.MAX_CO-1)*(Globals.MAX_FL -1)) + ((Globals.MAX_SP -1)* (Globals.MAX_SF -1)) +1];
private static int[] ListNodes= new int[((Globals.MAX_FR-1)*(Globals.MAX_CO-1)*(Globals.MAX_FL -1)) + ((Globals.MAX_SP -1)* (Globals.MAX_SF -1)) +1];
private static int[] Path1 = new int[((Globals.MAX_FR-1)*(Globals.MAX_CO-1)*(Globals.MAX_FL -1)) + ((Globals.MAX_SP -1)* (Globals.MAX_SF -1)) +1];
private static int[] Path2 = new int[((Globals.MAX_FR-1)*(Globals.MAX_CO-1)*(Globals.MAX_FL -1)) + ((Globals.MAX_SP -1)* (Globals.MAX_SF -1)) +1];
private static int Ptr1=0;
private static int Ptr2=0;
private static int StartNode;
private static int LastNode;
private static int recpnt=0; private static String output;
private static String violation;public CheckFileStructure_R3() {
super();
}
protected static void displayFileStructure() {
  output ="FILE STRUCTURE: As Extracted From The Current Example";
  FileStructure.reportFileStructure(output);
  FileStructure.display();
  output="FILE STRUCTURE display is DONE";
  FileStructure.reportFileStructure(output);
  output="==================================================================\n";
  FileStructure.reportFileStructure(output);
}
private static void displayInclusionMatrix() {
  output="INCLUSION MATRIX: Coordinates (x,y) represent FileNodeNumbers x and y\n";
  output=output + "\t" ;
  for (int i=1; i<=M;i++){
    output=output+ "\t"+i;
  }
  output=output+ "\n\n";
  for (int i=1; i<=M; i++){
    output=output+i+"\t";
    for (int j=1; j<=M;j++){
      output=output+"\t"+InclusionMatrix[i][j];
    }
    output=output+"\n";
  }
```

```
    output=output+"==========================================================================\n";
    reportInclusionMatrix(output);
  }
  private static void displayListNodes() {
    output="\n";
    output=output+"ListNodes[]:";
    for (int i=1; i<=Ptr; i++) {
      output=output+ "\t" + ListNodes[i];
    }
    output=output+"\n";
    reportOutput(output);
  }
  private static void displayLoop (int listSize) {
    output="\nVIOLATION(Predicate4_13): FileInclusionLOOP has been detected: \n";
    int lp=1;
    output=output+"\t";
    while (lp < listSize ) {
      output=output+ListNodes[lp]+ " |—> ";
      lp=lp+1;
    }
    output=output+ ListNodes[lp];
    output=output + "\n\tDETAILS:\n";   for (int lpp=1; lpp<=listSize; lpp++){
      output=output+"\t\tNodeNumber("+ListNodes[lpp]+")\t"+
        "FileName(" +ListFiles[ListNodes[lpp]].Name+"."+ListFiles[ListNodes[lpp]].Type+")\t"+
        "Framework("+ListFiles[ListNodes[lpp]].FR_Number+ ")\t"+
        "Component("+ListFiles[ListNodes[lpp]].FR_Number+","+ListFiles[ListNodes[lpp]].CO_Num-
ber+")\n";
    }
    reportViolation(output);
    return;
  }
  private static void displayMultiplePath (int Path1Size, int Path2Size ) {
    output="\nVIOLATION(Predicate4_12:OnlyIntermediateLevelMayHaveMultiplePaths)  has  been  de-
tected: ";
    int lp1=1;
    output=output+"\n\t";
    while (lp1 < Path1Size ) {
      output=output+Path1[lp1]+ " |—> ";
      lp1=lp1+1;
    }
    output=output+Path1[lp1];
    int lp2=1;
    output=output+"\n\t";
    while (lp2 < Path2Size ) {
      output=output+ Path2[lp2]+ " |—> ";
      lp2=lp2+1;
    }
```

```
    output=output+Path2[lp2];
    reportViolation (output);
    return;
  }
  private static void displayPath1() {
    output="\nPath1[]:";
    for (int i=1; i<=Ptr1; i++) {
      output=output+"\t"+Path1[i];
    }
    output=output+"\n";
    reportOutput(output);
  }
  private static void displayPath2() {
    output="\nPath2[]:";
    for (int i=1; i<=Ptr2; i++) {
      output=output+"\t"+Path2[i];
    }
    output=output+"\n";
    reportOutput(output);
  }
  private static void eXample_1() {
    FileStructure.initializeFor_Example_1();
  }
  private static void eXample_1_OLD() {
    FileStructure.initializeFor_Example_1();
  }
  private static void eXample_2() {
    FileStructure.initializeFor_Example_2();
  }
  private static void eXample_3_ACE() {
    FileStructure.initializeFor_Example_3();
  }
  private static void extractInclusionMatrix ( ) {
    // Calculate M: the dimension of the InclusionMatrix[MxM]
    M=0;
    for (int i=1; i<=FileStructure.F; i++){
      M=M + 3*FileStructure.C[i];
    }
    int ExtM=0;
    for (int i=1; i<=FileStructure.S; i++){
      ExtM=ExtM+FileStructure.SF[i];
    }
    M=M+ExtM; int tmpCount=0;
    for (int i=1; i<=FileStructure.F; i++){
      for (int j=1; j<=FileStructure.C[i]; j++){
        for (int k=1; k<=3; k++){
```

```
        int tmpFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNode-
Number;
        ListFiles[++tmpCount] = new tmpFileNode();
        ListFiles[tmpCount].NodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].ID.FileNodeNumber;
        ListFiles[tmpCount].Name=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
        ListFiles[tmpCount].Type=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
        ListFiles[tmpCount].FR_Number=i;
        ListFiles[tmpCount].CO_Number=j;
        int tmp=0;
        while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tmp]>0){
          int   tmpIncludedFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].In-
cludeList[tmp];
          if (tmpIncludedFileNodeNumber <= M) {
            InclusionMatrix[tmpFileNodeNumber][tmpIncludedFileNodeNumber]=1;
          }
          else {
            String      tmpFN=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName+"."+Fi-
leStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
            System.out.println("ERROR(Predicate4_13): File "+ tmpFN+ " with FileNode("+tmpFileNode-
Number+") from Framework("+i+") and Component("+i+","+j+
              ") includes FileNode("+tmpIncludedFileNodeNumber+") that has no existing file");
          }
        }
      }
    }
  } for (int i=1; i<=FileStructure.S; i++){
    for (int j=1; j<=FileStructure.SF[i]; j++){
      int tmpFileNodeNumber=FileStructure.CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber;

      ListFiles[++tmpCount] = new tmpFileNode();
      ListFiles[tmpCount].NodeNumber=       FileStructure.CO_Sh_Files[i].SharedFileList[j].ID.FileNode-
Number;
      ListFiles[tmpCount].Name=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileName;
      ListFiles[tmpCount].Type=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileType;
  int tmp=0;
      while (FileStructure.CO_Sh_Files[i].SharedFileList[j].IncludeList[++tmp] >0){
        int           tmpIncludedFileNodeNumber=FileStructure.CO_Sh_Files[i].SharedFileList[j].IncludeL-
ist[tmp];
        if (tmpIncludedFileNodeNumber <= M) {
          InclusionMatrix[tmpFileNodeNumber][tmpIncludedFileNodeNumber]=1;
        }
        else{
          String            tmpFN=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileName+"."+FileStruc-
ture.CO_Sh_Files[i].SharedFileList[j].FileType;
          System.out.println("ERROR(Predicate4_13): SharedFile "+ tmpFN+ " with FileNode("+tmpFile-
NodeNumber+") from SharedPool("+i+") includes FileNode("+tmpIncludedFileNodeNumber+") that has
```

```
no existing file");
            System.out.println("\tThis illegal FileNode will be ignored");
          }
        }
      }
    }       MatrixExtracted=true;
    return;
  }
  public static void initializeFileStructure() {
    if (Globals.EXAMPLE_NOM == 1) {
      eXample_1();
    }else {
      if (Globals.EXAMPLE_NOM==2){
        eXample_2();
      }else{
        if (Globals.EXAMPLE_NOM==3){
          eXample_3_ACE();
        }
        else{ System.out.println("Wrong Example Nomber.It has to be 1, 2, or 3");
          return;
        }
      }
    }
  }
  public static void main(String args[]) {
    Globals.EXAMPLE_NOM= Integer.parseInt(args[0]);
    initializeFileStructure();
    displayFileStructure(); predicate4_5_GoodFileStructure();
  }
  protected static boolean predicate4_10( ) {
    boolean returnValue=true;
    output="\nChecking for Predicate 4_10: ComponentsMainFilesShouldBeDistinct\n";
    for (int i=1; i<=FileStructure.F; i++){
      for (int j=1; j<=FileStructure.C[i]; j++){
        for (int k=1; k<=3; k++){
          String tmpName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          String tmpType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          String tmpFile=tmpName+"."+tmpType;
          for (int jj=1; jj<=FileStructure.C[i]; jj++){
            if (jj==j) { break;}
            for (int kk=1; kk<=3; kk++){
              String tmp2Name=FileStructure.CO_Fr_Files[i].ComponentList[jj].FileList[kk].FileName;
              String tmp2Type=FileStructure.CO_Fr_Files[i].ComponentList[jj].FileList[kk].FileType;
              String tmp2File=tmp2Name+"."+tmp2Type;
              if ((tmp2Name==tmpName)&&(tmp2Type==tmp2Type)){
                output=output+"\nVIOLATION(Predicate4_10): File "+tmpFile+" belongs to Componenet("+i+","+j+") and Component("+i+","+jj+") at the same time";
```

```
                returnValue=false;
              }
            }
          }
        }
      }
    }
    output=output+"\nChecking for Predicate 4_10 is complete ---- STATUS: "+ returnValue;
    output=output+"\n=================================================================\n";
    reportViolation(output);
    return returnValue;
  }
  protected static boolean predicate4_11 ( ) {
    reportViolation("\nChecking for Predicate 4_11: GoodUseOfFiles = predicates 2_14, 2_15, 2_16, and
  2_17");
    boolean returnValue1=predicate4_12();
    boolean returnValue2=predicate4_13();
    boolean returnValue3=predicate4_14();
    boolean returnValue4=predicate4_15();
    boolean returnValue=returnValue1 && returnValue2 && returnValue3 &&returnValue4;
    reportViolation("\nChecking for Predicate 4_11 is complete ---- STATUS: "+ returnValue);
    reportViolation("=================================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_12( ) {
    boolean returnValue=true;
    reportViolation("\nChecking for Predicate 4_12: InclusionLoopsMayNotExist");
    if (!MatrixExtracted){
      extractInclusionMatrix();
//    displayInclusionMatrix();
    }

    recursiveCheckForLoops(1,1);  if (LoopOccured) {returnValue=false;}

    reportViolation("\nChecking for Predicate 4_12 is complete ---- STATUS: "+ returnValue);
    reportViolation("=================================================================\n");
    return returnValue;
  }
  protected static boolean predicate4_13 ( ) {
    reportViolation("\nChecking for Predicate 4_13: GoodUseOf_top_Files = predicates 2_18, 2_19, and
  2_20");
    boolean returnValue1=predicate4_17();
    boolean returnValue2=predicate4_18();
    boolean returnValue3=predicate4_19();
    boolean returnValue=returnValue1 && returnValue2 && returnValue3;
    reportViolation("\nChecking for Predicate 4_13 is complete --- STATUS: "+ returnValue);
    reportViolation("=================================================================\n");
```

```
    return (returnValue);
  }
  protected static boolean predicate4_14 ( ) {
    reportViolation("\nChecking for Predicate 4_14: GoodUseOf_intermediate_Files = predicates 2_21 and
2_22");
    boolean returnValue=predicate4_20();
    reportViolation("\nChecking for Predicate 4_14 is complete —— STATUS: "+returnValue);
    reportViolation("==========================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_15 ( ) {
    reportViolation("\nChecking for Predicate 4_15: GoodUseOf_internal_Files = predicates 2_23 and
2_24");
    boolean returnValue1=predicate4_21();
    boolean returnValue2=predicate4_22();
    boolean returnValue=returnValue1&&returnValue2;
    reportViolation("\nChecking for Predicate 4_15 is complete —— STATUS: "+returnValue);
    reportViolation("==========================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_16 ( ) {
    reportViolation("\nChecking for Predicate 4_16: GoodUseOf_shared_Files = predicates 2_25 and 2_26");
    boolean returnValue=predicate4_23();
    reportViolation("\nChecking for Predicate 4_16 is complete —— STATUS: "+returnValue);
    reportViolation("==========================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_17 ( ) {
    boolean returnValue=true; reportViolation("\nChecking for predicate 4_17: NoFileMayInclu-
deA_top_File\n");
    tmpFileNode[] topNodeList = new tmpFileNode[Globals.MAX_FR * Globals.MAX_CO];
    int tmp=0;
    for (int i=1; i<=FileStructure.F; i++){
      for (int j=1; j<=FileStructure.C[i]; j++){
        for (int k=1; k<=3; k++){
          if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="top"){
            topNodeList[++tmp] = new tmpFileNode();
            topNodeList[tmp].NodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].ID.FileNodeNumber;
            topNodeList[tmp].Name=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
            topNodeList[tmp].Type=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
            topNodeList[tmp].CO_Number=j;
            topNodeList[tmp].FR_Number=i;
          }
        }
      }
```

```
  }
// Checking
  for (int ii=1; ii<=tmp; ii++){   // index ii for topNodeList[]
    for (int i=1; i<=FileStructure.F; i++){   // index i for frameowrks
      for (int j=1; j<=FileStructure.C[i]; j++){  // index j for components
        for (int k=1; k<=3; k++){  // index k for files
          int    tmpFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNo-
deNumber;
          String tmpFileName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          String tmpFileType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          int tmpFrameworkNumber=i;
          int tt=0;
          while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tt] >0){
            int          tmpIncludedFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].IncludeList[tt];
            if (tmpIncludedFileNodeNumber==topNodeList[ii].NodeNumber) {
              reportViolation("\nVIOLATION(Predicate4_18): Top File "+topNodeList[ii].Name+"."+top-
NodeList[ii].Type+
                "   from   Framework("+topNodeList[ii].FR_Number+")   is   included   by   "+tmpFile-
Name+"."+tmpFileType+
                " from Framwork("+tmpFrameworkNumber+")");
              returnValue=false;
            }
          }
        }
      }
    }
    for (int i=1; i<=FileStructure.S;i++){
      for (int j=1; j<=FileStructure.SF[i];j++){
        int tmpFileNodeNumber_sh=FileStructure.CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber;
        String tmpFileName_sh=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileName;
        String tmpFileType_sh=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileType;
        int tmpSharedPoolNumber=i;
        int tmpp=0;
        while (FileStructure.CO_Sh_Files[i].SharedFileList[j].IncludeList[++tmpp]>0){
          if   (FileStructure.CO_Sh_Files[i].SharedFileList[j].IncludeList[tmpp]==    topNodeList[ii].Node-
Number){
            reportViolation("\nVIOLATION(Predicate4_18): Top File "+topNodeList[ii].Name+"."+topNo-
deList[ii].Type+
              " from Framework("+topNodeList[ii].FR_Number+") is included by the shared file "+tmpFile-
Name_sh+"."+
              tmpFileType_sh+" from SharedPool("+tmpSharedPoolNumber+")");
            returnValue=false;
          }
        }
      }
    }
```

```
  } reportViolation("\nChecking for predicate 4_17 is complete —— STATUS: "+ returnValue);
  reportViolation("\n===================================================================\n");
  return returnValue;
}
protected static boolean predicate4_18 ( ) {
  boolean returnValue=true; reportViolation("\nChecking for predicate 4_18: FromSame_Fr_A_top_File-
MayIncludeOnlyIts_intermediate_File\n");
  for (int i=1;i<=FileStructure.F;i++){
    for (int j=1; j<=FileStructure.C[i];j++){
      for (int k=1; k<=3; k++){
        if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="top"){
          int    tmpFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNo-
deNumber;
          String tmpFileName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          String tmpFileType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          int tmpFrameworkNumber=i;
          int tmp=0;
          while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tmp]>0){
            int         tmpIncludedFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].IncludeList[tmp];
            boolean found2=false;
            for (int ii=1; ii<=FileStructure.F; ii++){
              if (found2) {break;}
              for (int jj=1; jj<=FileStructure.C[ii]; jj++){
                if (found2) {break;}
                for (int kk=1; kk<=3; kk++){
                  if (found2) {break;}
                  if        (FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].ID.FileNodeNum-
ber==tmpIncludedFileNodeNumber){
                    found2=true;
                    int        tmp2FileNodeNumber=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileL-
ist[kk].ID.FileNodeNumber;
                    String            tmp2FileName=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileL-
ist[kk].FileName;
                    String    tmp2FileType=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].Fi-
leType;
                    int tmp2FrameworkNumber=ii;
                    if (tmp2FrameworkNumber!=tmpFrameworkNumber){
                      reportViolation("\nVIOLATION(Predicate4_18): Top File "+tmpFileName+"."+tmpFi-
leType+" from Framework("+tmpFrameworkNumber+") includes File "+tmp2FileName+"."+tmp2File-
Type+" from Framework("+tmp2FrameworkNumber+")");
                      returnValue=false;
                      break;
                    }
                    if (tmp2FileNodeNumber==tmpFileNodeNumber){
                      reportViolation("\nVIOLATION(Predicate4_18): Top File "+tmpFileName+"."+tmpFi-
leType+" from Framework(" +tmpFrameworkNumber +") includes itself");
```

```
                    returnValue=false;
                    break;
                  }
                  if ((tmp2FrameworkNumber==tmpFrameworkNumber) &&
                    (tmp2FileName==tmpFileName) && (tmp2FileType=="inl")){
                    reportViolation("\nVIOLATION(Predicate4_18): Top File "+tmpFileName+"."+tmpFi-
leType+" from Framework(" +tmpFrameworkNumber +") includes its Internal File");
                    returnValue=false;
                    break;
                  }
                  if ((tmp2FrameworkNumber!=tmpFrameworkNumber) && (tmp2FileName !=tmpFile-
Name)){
                    reportViolation("\nVIOLATION(Predicate4_18): Top File "+tmpFileName+"."+tmpFi-
leType+" from Framework(" +tmpFrameworkNumber +") includes File "+tmp2FileName+"."+tmp2File-
Type+" from Framework("+tmp2FrameworkNumber+")");
                    returnValue=false;
                    break;
                  }
                }
              }
            }
          }
        }
      }
    }
  }
  }
  reportViolation("\nChecking for predicate 4_18 is complete — STATUS: "+ returnValue);
  reportViolation("\n=========================================================\n");
  return returnValue;
}
protected static boolean predicate4_19 ( ) {
  boolean returnValue=true; reportViolation("\nChecking for predicate 4_19: A_top_FileMayIn-
clude_shared_FilesFromPermittedSharedPools\n");
  tmpFileNode[] sharedNodeList = new tmpFileNode[Globals.MAX_SP * Globals.MAX_SF];
  int tmp=0;
  for (int i=1; i<=FileStructure.S; i++){
    for (int j=1; j<=FileStructure.SF[i]; j++){
      if (FileStructure.CO_Sh_Files[i].SharedFileList[j].FileType=="s"){
        sharedNodeList[++tmp] = new tmpFileNode();
        sharedNodeList[tmp].NodeNumber=FileStructure.CO_Sh_Files[i].SharedFileList[j].ID.FileNode-
Number;
        sharedNodeList[tmp].Name=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileName;
        sharedNodeList[tmp].Type=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileType;
        sharedNodeList[tmp].CO_Number=j;
        sharedNodeList[tmp].FR_Number=i;
      }
```

```
    }
  }
  int NumberOfSharedFiles=tmp;
// Checking
  for (int i=1; i<=FileStructure.F; i++){
    for (int j=1; j<=FileStructure.C[i]; j++){
      for (int k=1; k<=3; k++){
        if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="top"){
          int    tmpFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNo-
deNumber;
          String tmpFileName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          String tmpFileType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          int tmpFrameworkNumber=i;
          int tmpp=0;
          while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tmpp]>0){
            // find out if this file node is a shared one
            int      tmpSharedNode_check=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].Inclu-
deList[tmpp];
            int tt=0;
            boolean found=false;
            while (tt<= tmp){
              if (found) {break;}
              tt=tt+1; if (tt>tmp) {break;}
              if (sharedNodeList[tt].NodeNumber==tmpSharedNode_check){
                found=true;
                // check if tmpFrameworkNumber belongs to the permitted set of Frameworks to access the
SharedPool of the detected sharedfile
                int tmpSharedPoolNumber=sharedNodeList[tt].FR_Number;
                int tmppp=0;
                boolean found2=false;
                while                    (FileStructure.CO_Sh_Files[tmpSharedPoolNumber].PermittedFrame-
works[++tmppp]>0){
                  if (found2) {break;}
                  if                     (FileStructure.CO_Sh_Files[tmpSharedPoolNumber].PermittedFrame-
works[tmppp]==tmpFrameworkNumber){
                    found2=true;
                  }
                }
                if (!found2){
                  reportViolation("VIOLATION(Predicate4_19):  Top  File  "+tmpFileName+"."+tmpFile-
Type+" from Framework("+tmpFrameworkNumber+") includes a shared file from a non-accessible Shared-
Pool("+tmpSharedPoolNumber+")");
                  returnValue=false;
                }
              }
            }
```

```
          }
        }
      }
    }
  }
  reportViolation("\nChecking for predicate 4_19 is complete — STATUS: "+ returnValue);
  reportViolation("\n=======================================================================\n");
  return returnValue;
}
protected static boolean predicate4_20 ( ) {
  boolean returnValue=true; reportViolation("\nChecking for predicate 4_20: An_intermediate_fileMayIn-
cludeOnlyIts_internal_FileAndMayIncludeOther_intermediate_FilesFromSame_Fr\n");
  for (int i=1; i<=FileStructure.F; i++){
    for (int j=1; j<=FileStructure.C[i]; j++){
      for (int k=1; k<=3; k++){
        if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="inm"){
          int    tmpFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNo-
deNumber;
          int tmpFrameworkNumber=i;
          String tmpFileName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          String tmpFileType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          int tmp=0;
          while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tmp]>0){
            int          tmpIncludedFileNodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].IncludeList[tmp];
            boolean found=false;
            for (int ii=1; ii<=FileStructure.F;ii++){
              if (found) { break;}
              for (int jj=1; jj<=FileStructure.C[ii]; jj++){
                if (found) { break;}
                for (int kk=1; kk<=3; kk++) {
                  if (found) { break;}
                  int          tmp2FileNodeNumber=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileL-
ist[kk].ID.FileNodeNumber;
                  int tmp2FrameworkNumber=ii;
                  String   tmp2FileName=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].File-
Name;
                  String    tmp2FileType=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].File-
Type;

                  if (tmp2FileNodeNumber==tmpIncludedFileNodeNumber){
                    found=true;
                    if (tmp2FileNodeNumber==tmpFileNodeNumber){
                      reportViolation("\nVIOLATION(Predicate4_20):   Intermediate   File   "+tmpFile-
Name+"."+tmpFileType+" from Framework("+tmpFrameworkNumber+") includes itself");
                      returnValue=false;
                    }
```

```
                    if ((tmp2FileName==tmpFileName) && (tmp2FrameworkNumber==tmpFramework-
Number) && (tmp2FileType=="top")){
                      reportViolation("\nVIOLATION(Predicate4_20): Intermediate File "+tmpFile-
Name+"."+tmpFileType+" from Framework("+tmpFrameworkNumber+") includes its top file "+tmp2File-
Name+"."+tmp2FileType);
                      returnValue=false;
                    }
                    if ((tmp2FrameworkNumber==tmpFrameworkNumber) && (tmp2FileName!=tmpFile-
Name) && (tmp2FileType=="inl")){
                      reportViolation("\nVIOLATION(Predicate4_20): Intermediate File "+tmpFile-
Name+"."+tmpFileType+" from Framework("+tmpFrameworkNumber+") includes the non-intermediate
"+tmp2FileName+"."+tmp2FileType+" from the same framework");
                      returnValue=false;
                    }
                    if (tmp2FrameworkNumber!=tmpFrameworkNumber){
                      reportViolation("\nVIOLATION(Predicate4_20): Intermediate File "+tmpFile-
Name+"."+tmpFileType+" from Framework("+tmpFrameworkNumber+" includes "+tmp2File-
Name+"."+tmp2FileType+" from Framework("+tmp2FrameworkNumber+")");
                      returnValue=false;
                    }
                  }
                }
              }
            }
            if (!found) {
              boolean found_sh=false;
              for (int ii=1; ii<=FileStructure.S; ii++){
                if (found_sh) { break;}
                for (int jj=1; jj<=FileStructure.SF[ii]; jj++){
                  if (found_sh) {break;}
                  if (FileStructure.CO_Sh_Files[ii].SharedFileList[jj].ID.FileNodeNumber== tmpIncluded-
FileNodeNumber){
                    found_sh=true;
                    String tmpFileName_sh=FileStructure.CO_Sh_Files[ii].SharedFileList[jj].FileName;
                    String tmpFileType_sh=FileStructure.CO_Sh_Files[ii].SharedFileList[jj].FileType;
                    reportViolation("\nVIOLATION(Predicate4_20): Intermediate file "+tmpFile-
Name+"."+tmpFileType+
                      " from Framework("+tmpFrameworkNumber+") includes "+tmpFile-
Name_sh+"."+tmpFileType_sh+
                      " from SharedPool("+ii+")");
                    returnValue=false;
                  }
                }
              }
              if(!found_sh){
                reportViolation("\nERROR(Predicate4_20): Intermediate File "+tmpFileName+"."+tmpFi-
leType+" includes FileNode("+tmpIncludedFileNodeNumber+") that is not found in any Framework or Sha-
```

```
redPool");
                returnValue=false;
              }
            }
          }
        }
      }
    }
  }

  reportViolation("\nChecking for predicate 4_20 is complete — STATUS: "+ returnValue);
  reportViolation("\n==========================================================================\n");
  return returnValue;
 }
 protected static boolean predicate4_21 ( ) {
  boolean returnValue=true;
  reportViolation("\nChecking for predicate 4_21: An_internal_FileMayNotIncludeAnyFile");
  tmpFileNode[] InternalNodeList = new tmpFileNode[Globals.MAX_FR * Globals.MAX_CO];
// Checking
  for (int i=1; i<=FileStructure.F; i++){      // index i for frameowrks
    for (int j=1; j<=FileStructure.C[i]; j++){  // index j for components
      for (int k=1; k<=3; k++){  // index k for files
        if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="inl") {
          int tt=0;
          while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tt] >0){
            reportViolation("\nVIOLATION(Predicate 4_21): InternalFile "+ FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName+"."+FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType+" Includes FileNode "+FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[tt]);
            returnValue=false;
          }
        }
      }
    }
  }

  reportViolation("\nChecking for predicate 4_21 is complete — STATUS:"+returnValue);
  return returnValue;
}
protected static boolean predicate4_22 ( ) {
  boolean returnValue=true;
  reportViolation("\nChecking for predicate 4_22: NoFileMayIncludeAn_Internal_FileExceptIts_intermediate_File\n");
  tmpFileNode[] InternalNodeList = new tmpFileNode[Globals.MAX_FR * Globals.MAX_CO];
  int tmp=0;
  for (int i=1; i<=FileStructure.F; i++){
    for (int j=1; j<=FileStructure.C[i]; j++){
```

```
        for (int k=1; k<=3; k++){
          if (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType=="inl"){
            InternalNodeList[++tmp] = new tmpFileNode();
            InternalNodeList[tmp].NodeNumber=FileStructure.CO_Fr_Files[i].ComponentList[j].FileL-
ist[k].ID.FileNodeNumber;
            InternalNodeList[tmp].Name=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].File-
Name;
            InternalNodeList[tmp].Type=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].File-
Type;
            InternalNodeList[tmp].CO_Number=j;
            InternalNodeList[tmp].FR_Number=i;
          }
        }
      }
    }
  // Checking
    for (int ii=1; ii<=tmp; ii++){     // index ii for InternalNodeList[]
      for (int i=1; i<=FileStructure.F; i++){      // index i for frameowrks
        for (int j=1; j<=FileStructure.C[i]; j++){ // index j for components
          for (int k=1; k<=3; k++){ // index k for files
            int tt=0;
            while (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[++tt] >0){
              if      ((FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[tt]==InternalNo-
deList[ii].NodeNumber) &&
                    ((FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName   !=InternalNodeL-
ist[ii].Name) ||
                    ((FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName == InternalNodeL-
ist[ii].Name) &&
                  (FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType != "inm")))){
                    reportViolation("\nVIOLATION(Predicate 4_22): Internal File "+InternalNodeL-
ist[ii].Name+"."+InternalNodeList[ii].Type+" is included by "+FileStructure.CO_Fr_Files[i].Componen-
tList[j].FileList[k].FileName+"."+FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType+"
from Framework("+i+") — Component("+i+","+j+")");
                    returnValue=false;
              }
            }
          }
        }
      }
    }

  reportViolation("\nChecking for predicate 4_22 is complete —— STATUS:"+returnValue);
  return returnValue;
}
protected static boolean predicate4_23 ( ) {
  output="\nChecking for Predicate4_23: A_shared_FileMayIncludeOnluOther_shared_FilesFrom-
Same_SP\n";
```

```
    boolean returnValue=true;
    for (int i=1; i<=FileStructure.S; i++){
      // extract the shared files list of FileNodes for the ith shared pool
      int tmp=0;
      int[] tmpNodeList=new int[FileStructure.SF[i]+1];
      for (int j=1; j<=FileStructure.SF[i]; j++) {
        tmpNodeList[++tmp]=FileStructure.CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber;
      }
      reportViolation("");
      for (int sf=1;sf<=FileStructure.SF[i];sf++){
        int tt=0;
        while (FileStructure.CO_Sh_Files[i].SharedFileList[sf].IncludeList[++tt]>0){
          int ttt=1;
          while ((ttt<=tmp) && (FileStructure.CO_Sh_Files[i].SharedFileList[sf].IncludeList[tt]!=tmpNo-
deList[ttt])){ttt=ttt+1;}
          if (ttt>tmp) {
            output=output+"\nVIOLATION(Predicate4_23):  FileNode("+FileStructure.CO_Sh_Files[i].Sha-
redFileList[sf].ID.FileNodeNumber+
                ") from SharedPool("+i+") includes FileNode("+FileStructure.CO_Sh_Files[i].SharedFileL-
ist[sf].IncludeList[tt]+
                ") from another SharedPool or Framework";
            returnValue=false;
          }
        }
      }
    } output=output+"\nChecking for Predicate4_23 is complete —— STATUS: "+returnValue;
    output=output+"\n================================================================\n";
    reportViolation(output);
    return returnValue;
  }
  protected static boolean predicate4_5_GoodFileStructure ( ) {
    reportViolation("\nChecking for GoodFileStructure_predicate4_5: GoodFileStructure = predicates 2_6,
2_7, and 2_8");
    boolean returnValue1=predicate4_6();
    boolean returnValue2=predicate4_7();
    boolean returnValue3=predicate4_8();
    boolean returnValue=returnValue1 && returnValue2 && returnValue3;
    reportViolation("\nChecking for GoodFileStructure_predicate4_5 is complete —— STATUS:"+returnVa-
lue);
    reportViolation("================================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_6( ) {
    boolean returnValue=true;
    reportViolation("\nChecking for Predicate 4_6: OnlyFourFileCategories\n");
    for (int i=1; i<=FileStructure.F; i++){
      for (int j=1; j<=FileStructure.C[i]; j++){
```

```
        for (int k=1; k<=3; k++){
          String tmpType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
          String tmpName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
          if ((tmpType!="top") && (tmpType!="inm") && (tmpType!="inl")){
            reportViolation("\nVIOLATION(Predicate4_6): File "+tmpName+"."+tmpType+ " from Frame-
work("+i+") uses an invlaid File Extension");
            returnValue=false;
          }
        }
      }
    }
      for (int i=1; i<=FileStructure.S; i++){
      for (int j=1; j<=FileStructure.SF[i]; j++){
        String tmpType=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileType;
        String tmpName=FileStructure.CO_Sh_Files[i].SharedFileList[j].FileName;
        if (tmpType!="s"){
          reportViolation("\nVIOLATION(Predicate4_6): File "+tmpName+"."+tmpType+ " from Shared-
Pool("+i+") uses an invlaid File Extension");
          returnValue=false;
        }
      }
    }

    reportViolation("\nChecking for Predicate 4_6 is complete —— STATUS: "+returnValue);
    reportViolation("\n==========================================================================\n");
    return returnValue;
  }
  protected static boolean predicate4_7 ( ) {
    reportViolation("\nChecking for Predicate 4_7: GoodDistinctionOfFiles= Predicates 2_9, and 2_10");
    boolean returnValue1=predicate4_9();
    boolean returnValue2=predicate4_10();
    boolean returnValue=returnValue1&&returnValue2;
    reportViolation("\nChecking for Predicate 4_7 is complete —— STATUS: "+returnValue);
    reportViolation("==========================================================================\n");
    return (returnValue);
  }
  protected static boolean predicate4_8 ( ) {
    reportViolation("\nChecking for Predicate 4_8: GoodFileInclusionRelation = predicates 2_11, 2_12, and
2_13");
    boolean returnValue1=predicate4_11();
    boolean returnValue2=predicate4_12();
    boolean returnValue=returnValue1 && returnValue2;
    reportViolation("\nChecking for Predicate 4_8 is complete —— STATUS: "+returnValue);
    reportViolation("==========================================================================\n");
    return (returnValue);
  }
```

```
protected static boolean predicate4_9( ) {
  boolean returnValue=true;
  reportViolation("\nChecking for Predicate 4_9: Fr_SetsOfFilesShouldBeDistinct\n");
  for (int i=1; i<=FileStructure.F; i++){
    for (int j=1; j<=FileStructure.C[i]; j++){
      for (int k=1; k<=3; k++){
        String tmpName=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileName;
        String tmpType=FileStructure.CO_Fr_Files[i].ComponentList[j].FileList[k].FileType;
        String tmpFile=tmpName+"."+tmpType;
        for (int ii=1; ii<=FileStructure.F; ii++){
          for (int jj=1; jj<=FileStructure.C[ii]; jj++){
            if ((ii==i) && (jj==j)) {break;}
            for (int kk=1; kk<=3; kk++){
              String tmp2Name=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].FileName;
              String tmp2Type=FileStructure.CO_Fr_Files[ii].ComponentList[jj].FileList[kk].FileType;
              String tmp2File=tmp2Name+"."+tmp2Type;
              if ((tmp2Name==tmpName)&&(tmp2Type==tmp2Type)){
                reportViolation("\nVIOLATION(Predicate4_9): File "+
                  tmpFile+" belongs to Framework("+i+") and Framework("
                    +ii+") at the same time");
                returnValue=false;
              }
            }
          }
        }
      }
    }
  }

  reportViolation("\nChecking for Predicate 4_9 is complete —— STATUS: "+returnValue);
  reportViolation("\n=========================================================\n");
  return returnValue;
}
private static void recursiveCheckForLoops (int x, int y ) {
  if (x>M) {return;}
  if (x==y){y=y+1;} if ((x>=M) && (y>M)) {return;}
  while((y<=M) && (InclusionMatrix[x][y]!=1)){y=y+1;}
  if (x==y) {
    y=y+1;
    while((y<=M) && (InclusionMatrix[x][y]==0)){y=y+1;}
  }
  if (y>M){
    if ((Ptr==0)&&(x<=M)){
      recursiveCheckForLoops(x+1,1);
    }
    else{
      Ptr=Ptr-1;
```

```
        while ((Ptr>0)&& (ListNodes[Ptr+1]+1 > M)){Ptr=Ptr-1;}
        if (Ptr>0){
          if(x<=M){
            recursiveCheckForLoops(ListNodes[Ptr], ListNodes[Ptr+1]+1);
          }
        }
        else{
          if(x<=M){
            recursiveCheckForLoops(ListNodes[Ptr+1]+1, 1);
          }
        }
      }
    }
    else{
      if (Ptr!=1){ListNodes[++Ptr]=x; ListNodes[++Ptr]=y;}
      else { ListNodes[++Ptr]=y;}
      // compress
      if (Ptr>=4){
        if(ListNodes[Ptr-1]==ListNodes[Ptr-2]){
          ListNodes[Ptr-1]=ListNodes[Ptr];
          Ptr=Ptr-1;
        }
        // cleanup
        int tmpPtr=Ptr-1;
        while ((tmpPtr>1) && (ListNodes[Ptr]!=ListNodes[tmpPtr])) {tmpPtr=tmpPtr-1;}
        if (tmpPtr>1){
          Ptr=Ptr-1;
          if ((y<=M) && (x<=M)){
            recursiveCheckForLoops(ListNodes[Ptr], ListNodes[Ptr+1]+1);
          }
        }
      }    //check for loops
      if (Ptr>=2){
        if(ListNodes[Ptr]==ListNodes[1]){
          LoopOccured=true;
          displayLoop(Ptr);
          Ptr=Ptr-1;
          if (x<=M){
            recursiveCheckForLoops(ListNodes[Ptr], ListNodes[Ptr+1]+1);
          }
        }
        else{
          if ((y<=M)&&(x<=M)){
            recursiveCheckForLoops(y,1);
          }
        }
      }
```

```
    }
  }
  private static void recursiveCheckForMultiplePaths (int x, int y, int pathID ) {
    if (x>M) {return;}  if ((x>=M) && (y>M)) {return;}
    while(((y<=M)&&(InclusionMatrix[x][y]==0))||(x==y)){y=y+1;}
    if (pathID==1) {
      if (y<=M){
        if (Ptr1==0) { StartNode=x;}
        Path1[++Ptr1]=x;
        Path1[++Ptr1]=y;
        //Compress
        if (Ptr1>2){
          if (Path1[Ptr1-1]==Path1[Ptr1-2]){
            Ptr1=Ptr1-1;
            Path1[Ptr1]=Path1[Ptr1+1];
          }
          //clean up any redundancies as well as skip any loops
          int tmpPtr1=Ptr1-1;
          while ((tmpPtr1>=1) && (Path1[Ptr1]!=Path1[tmpPtr1])){tmpPtr1=tmpPtr1-1;}
          if (tmpPtr1>=1){Ptr1=Ptr1-1;}
        }
        if (Ptr1==2){
          recpnt=1;
          recursiveCheckForMultiplePaths(Path1[Ptr1], Path1[Ptr1+1]+1,1);
        }
        if (Ptr1<2){
          StartNode=StartNode+1;
          recpnt=2;
          if(StartNode<=M){recursiveCheckForMultiplePaths(StartNode,1,1);}
          else{return;}
        }
        if (Ptr1>2){ // Path1 has more than two nodes ... ready .. move to Path2
          if (StartNode<=M){
            recpnt=3;
            recursiveCheckForMultiplePaths(StartNode,1,2);
            recpnt=32;
            recursiveCheckForMultiplePaths(Path1[Ptr1],Path1[Ptr1+1]+1,1);
          }
          else{return;}

        }
      }
      else{ // pathID=1 && Y>M
        if (Ptr1>=2){
          Ptr1=Ptr1-1;
          recpnt=4;
          recursiveCheckForMultiplePaths(Path1[Ptr1],Path1[Ptr1+1]+1,1);
```

```
        }
        else{ // Ptr1<2
          StartNode=StartNode+1;
          Ptr1=0;
          recpnt=5;
          if (StartNode<=M){recursiveCheckForMultiplePaths(StartNode,1,1);}
          else{return;}
        }
      }
    }
    else{ // pathID=2
      if(y<=M){
        Path2[++Ptr2]=x;
        Path2[++Ptr2]=y;
        //compress
        if(Ptr2>2){
          if(Path2[Ptr2-1]==Path2[Ptr2-2]){
            Ptr2=Ptr2-1;
            Path2[Ptr2]=Path2[Ptr2+1];
          }
          //clean up and skip any loops
          int tmpPtr2=Ptr2-1;
          while ((tmpPtr2>=1) && (Path2[Ptr2]!=Path2[tmpPtr2])){tmpPtr2=tmpPtr2-1;}
          if (tmpPtr2>=1){Ptr2=Ptr2-1;}
        }
        // ensure that Path2 and Path1 are distinct in the middle
        if(Ptr2>=2){
          if(Path2[2]==Path1[2]){
            Ptr2=1;
            recpnt=6;
            recursiveCheckForMultiplePaths(Path2[Ptr2],Path2[Ptr2+1]+1,2);
          }
        }
        //check for multiple paths
        if (Ptr2>=2){
          if((Path1[1]==Path2[1])&&(Path1[Ptr1]==Path2[Ptr2])){
            MultiplePathOccured=true;
            displayMultiplePath(Ptr1,Ptr2);
            Ptr2=Ptr2-1;
          }
          recpnt=7;
          recursiveCheckForMultiplePaths(Path2[Ptr2],Path2[Ptr2+1]+1,2);
        }
        // forward
        if(Ptr2<2){
          Ptr2=0;
//        StartNode=StartNode+1;
```

```
//      Ptr1=0;
//      recpnt=8;
//      if (StartNode<=M){
//        recursiveCheckForMultiplePaths(StartNode,1,1);
//      }
//      else{
          return;
//      }
      }
    }
    else{ // pathID=2 & y>m
      if(Ptr2>0){Ptr2=Ptr2-1;}
      if(Ptr2>0){
        recpnt=9;
        recursiveCheckForMultiplePaths(Path2[Ptr2], Path2[Ptr2+1]+1,2);
      }
      else{ // Ptr2=<0
        Ptr2=0;
//      StartNode=StartNode+1;
//      Ptr1=0;
//      recpnt=10;
//      if (StartNode<=M){
//        recursiveCheckForMultiplePaths(StartNode,1,1);
//      }
//      else{
          return;
//      }
      }
    }
  }
}
public static void reportInclusionMatrix(String output){
  System.out.println("\n"+output+"\n");
  return;
}
public static void reportOutput(String output){
  System.out.println("\n"+output+"\n");
  return;
}
public static void reportViolation(String output){
  System.out.println("\n"+output+"\n");
  return;
}
}
```

### A1.2.2 Globals.java

This file contains the class that is used to provide all global variables used in the implementation of this prototype. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */

public class Globals {
  public static int EXAMPLE_NOM;    // number of the example
  public static int MAX_FR=20;    // maximum number of frameworks -1
  public static int MAX_CO=50;    // maximum number of components -1 per framework
  public static int MAX_FL=4;    // maximum number of files -1 per component
  public static int MAX_SP=3;    // maximum number of shared pools -1
  public static int MAX_SF=30;  // maximum number of shared files per shared pool -1
  public static int MAX_INCL= 20; // maximum number of of files that can be included by a single included
files
  public static int NodeCount=0;
public Globals() { }
}
```

### A1.2.3 tmpFileNode.java

This file contains the class that is used to represent an internal temporary file node. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
```

```
public class tmpFileNode {
  int NodeNumber;
  String Name;
  String Type;
  int CO_Number;
  int FR_Number;
  public tmpFileNode() { }
}
```

### A1.2.4 FileID.java

This file contains the class that is used to represent a file ID. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class FileID {
  int ParentFrameworkNodeNumber;
  int ParentComponentNodeNumber;
  int FileNodeNumber;  // FileNodeNumber must be unique
// — no other file (whether shared or not) can have the same number
public FileID() { super(); }
}
```

### A1.2.5 FileNode.java

This file contains the class that is used to represent a FileNode in the File Structure. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
```

```
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class FileNode {
  FileID ID=new FileID();
  String FileName= null;
  String FileType= null;
  int IncludeList[]=new int[Globals.MAX_INCL];
  public FileNode() {super();}
}
```

### A1.2.6 ComponentID.java

This file contains the class that is used to represent a component ID. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class ComponentID {
  int ParentFrameworkNodeNumber;
  int ComponentNodeNumber;   // this must be a unique number for each component/**
  public ComponentID() { }
}
```

### A1.2.7 ComponentNode.java

This file contains the class that is used to represent a component node. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
```

```
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class ComponentNode {  ComponentID ID = new ComponentID();
  FileNode FileList[]=new FileNode[Globals.MAX_FL];
  public ComponentNode() {super();}
}
```

### A1.2.8 FrameworkNode.java

This file contains the class that is used to represent a framework node. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class FrameworkNode {
  int FrameworkNodeNumber;
  // this must be a unique number for each framework
  ComponentNode ComponentList[]=new ComponentNode[Globals.MAX_CO];
  public FrameworkNode() {super();}
}
```

### A1.2.9 SharedFileID.java

This file contains the class that is used to represent a shared file ID. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
```

```
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class SharedFileID {
  int ParentSharedPoolNodeNumber;
  int FileNodeNumber;        // FileNodeNumber must be unique
    // — no other file (whether shared or not) can have the same number
  public SharedFileID() {super();}
}
```

### A1.2.10 SharedFileNode.java

This file contains the class that is used to represent a shared file node in the File Structure. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class SharedFileNode {
  SharedFileID ID=new SharedFileID();
  String FileName= null;
  String FileType= null;
  int IncludeList[]=new int[Globals.MAX_INCL];
  public SharedFileNode() {super();}
}
```

### A1.2.11 SharedPoolNode.java

This file contains the class that is used to represent a shared file node. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
```

```
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */
public class SharedPoolNode {
  int SharedPoolNodeNumber;
  SharedFileNode[] SharedFileList = new SharedFileNode[Globals.MAX_SF];
  int[] PermittedFrameworks = new int[Globals.MAX_FR];
  public SharedPoolNode() {super();}
}
```

### A1.2.12 FileStructure.java

This file contains the implementation of the File Structure. Users should notice that the three examples are imbedded in this class. Next is the source code listing for this file:

```
package V1_FileStructure_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#1 For CO_Toolkits File Structures — Chapter 4 of My Ph.D. Thesis
 */

public class FileStructure {
  public static int F;          // NumberOfFrameworks;
  public static int[] C = new int[Globals.MAX_FR];  // to hold number of components per framework
  public static int S;          // NumberOfSharedPools;
  public static int[] SF = new int[Globals.MAX_SP]; // to hold number of shared files per shared pool
  private static String output;
  public static FrameworkNode[] CO_Fr_Files=new FrameworkNode[Globals.MAX_FR];
  public static SharedPoolNode[] CO_Sh_Files = new SharedPoolNode[Globals.MAX_SP];
  public static void display() {
    displayFrameworks();
    displaySharedPools();
  }
public static void displayFrameworks() {
  output="";
  for (int i=1; i<= F; i++ ) {
   output=output+"\n\nFramework("+i+")\n";
```

```
    for (int j=1; j<= C[i]; j++ ){
    output=output+"\tComponent("+i+","+j+")\n";
    for (int k=1; k<= 3; k++){
      output=output+"\t\tFileNode("+        CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNodeNum-
ber+"):\t"+
        CO_Fr_Files[i].ComponentList[j].FileList[k].FileName+"."+              CO_Fr_Files[i].Componen-
tList[j].FileList[k].FileType;
      output=output+"\tIncludes FileNodes: " ;
      int tmp=1;
      while (CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[tmp] >0) {
        output=output+"\t"+ CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[tmp++];
      }
      output=output+"\n";
      }
    }
  }
  reportFileStructure(output);
}
public static void displaySharedPools() {
  output="";
  for (int i=1; i<= S; i++ ) {
   output=output+"\nSharedPool("+i+")";
   for (int j=1; j<= SF[i]; j++ ){
     output=output+"\n\t\tFileNode("+       CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber+"):\t"+
CO_Sh_Files[i].SharedFileList[j].FileName+"."+ CO_Sh_Files[i].SharedFileList[j].FileType;
     output=output+"\tIncludes FileNodes: ";
     int tmp=1;
     while (CO_Sh_Files[i].SharedFileList[j].IncludeList[tmp] >0) {
       output=output+"\t"+ CO_Sh_Files[i].SharedFileList[j].IncludeList[tmp++];
     }
   }
   output=output+"\n\tPermitted Frameworks to this SharedPool are: ";
   int tttt=0;
   while (CO_Sh_Files[i].PermittedFrameworks[++tttt]>0) {
     output=output+"\tFramework("+CO_Sh_Files[i].PermittedFrameworks[tttt]+")";
   }
   output=output+"\n";
  }
  reportFileStructure(output);
}
private static void frameworks_Structure_Example_1() {
  F=2;
  C[1]=2;
  C[2]=2;// initialize the array structure ... new array then go over each element and initialize it.
  for (int i=1; i<=F; i++){
    CO_Fr_Files[i] = new FrameworkNode();
    CO_Fr_Files[i].FrameworkNodeNumber=i;
```

```
      for (int j=1; j<=C[i]; j++){
        CO_Fr_Files[i].ComponentList[j]=new ComponentNode();
        CO_Fr_Files[i].ComponentList[j].ID.ComponentNodeNumber=j;
        CO_Fr_Files[i].ComponentList[j].ID.ParentFrameworkNodeNumber=i;
        for (int k=1; k<=3; k++){
          CO_Fr_Files[i].ComponentList[j].FileList[k]=new FileNode();
          CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNodeNumber=++Globals.NodeCount;
          CO_Fr_Files[i].ComponentList[j].FileList[k].ID.ParentComponentNodeNumber=j;
        }
      }
    }
 // File names Assingments
 // Component(1,1)
   CO_Fr_Files[1].ComponentList[1].FileList[1].FileName="file11";
   CO_Fr_Files[1].ComponentList[1].FileList[1].FileType="top";
   CO_Fr_Files[1].ComponentList[1].FileList[2].FileName="file11";
   CO_Fr_Files[1].ComponentList[1].FileList[2].FileType="inm";
   CO_Fr_Files[1].ComponentList[1].FileList[3].FileName="file11";
   CO_Fr_Files[1].ComponentList[1].FileList[3].FileType="inl";
 // Component (1,2)
   CO_Fr_Files[1].ComponentList[2].FileList[1].FileName="file12";
   CO_Fr_Files[1].ComponentList[2].FileList[1].FileType="top";
   CO_Fr_Files[1].ComponentList[2].FileList[2].FileName="file12";
   CO_Fr_Files[1].ComponentList[2].FileList[2].FileType="inm";
   CO_Fr_Files[1].ComponentList[2].FileList[3].FileName="file12";
   CO_Fr_Files[1].ComponentList[2].FileList[3].FileType="inl";
 // Component (2,1)
   CO_Fr_Files[2].ComponentList[1].FileList[1].FileName="file21";
   CO_Fr_Files[2].ComponentList[1].FileList[1].FileType="top";
   CO_Fr_Files[2].ComponentList[1].FileList[2].FileName="file21";
   CO_Fr_Files[2].ComponentList[1].FileList[2].FileType="inm";
   CO_Fr_Files[2].ComponentList[1].FileList[3].FileName="file21";
   CO_Fr_Files[2].ComponentList[1].FileList[3].FileType="inl";
 // Component (2,2)
   CO_Fr_Files[2].ComponentList[2].FileList[1].FileName="file22";
   CO_Fr_Files[2].ComponentList[2].FileList[1].FileType="top";
   CO_Fr_Files[2].ComponentList[2].FileList[2].FileName="file22";
   CO_Fr_Files[2].ComponentList[2].FileList[2].FileType="inm";
   CO_Fr_Files[2].ComponentList[2].FileList[3].FileName="file22";
   CO_Fr_Files[2].ComponentList[2].FileList[3].FileType="inl";
 //Assign Include lists
 //Component (1,1)  CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[1]=2;
   CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[2]=13;
   CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[3]=15;
   CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[4]=-1;
   CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[1]=3;
   CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[2]=5;
```

```
  CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[3]=-1;
  CO_Fr_Files[1].ComponentList[1].FileList[3].IncludeList[1]=-1;
//Component (1,2)  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[1]=5;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[2]=-1;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[1]=6;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[2]=-1;
  CO_Fr_Files[1].ComponentList[2].FileList[3].IncludeList[1]=-1;
//Component (2,1)  CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[1]=8;
  CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[2]=-1;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[1]=9;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[2]=-1;
  CO_Fr_Files[2].ComponentList[1].FileList[3].IncludeList[1]=-1;
//Component (2,2)  CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[1]=11;
  CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[2]=13;
  CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[3]=14;
  CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[1]=12;
  CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[2]=8;
  CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[3]=-1;
  CO_Fr_Files[2].ComponentList[2].FileList[3].IncludeList[1]=-1;
}
private static void frameworks_Structure_Example_2() {
  F=3;
  C[1]=1;
  C[2]=1;
  C[3]=2;
// initialize the array structure ... new array then go over each element and initialize it.
  for (int i=1; i<=F; i++){
    CO_Fr_Files[i] = new FrameworkNode();
    CO_Fr_Files[i].FrameworkNodeNumber=i;
    for (int j=1; j<=C[i]; j++){
      CO_Fr_Files[i].ComponentList[j]= new ComponentNode();
      CO_Fr_Files[i].ComponentList[j].ID.ComponentNodeNumber=j;
      CO_Fr_Files[i].ComponentList[j].ID.ParentFrameworkNodeNumber=i;
      for (int k=1; k<=3; k++){
        CO_Fr_Files[i].ComponentList[j].FileList[k]=new FileNode();
        CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNodeNumber=++Globals.NodeCount;
        CO_Fr_Files[i].ComponentList[j].FileList[k].ID.ParentComponentNodeNumber=j;
      }
    }
  }
// File names Assingments
// Component(1,1)
  CO_Fr_Files[1].ComponentList[1].FileList[1].FileName="file11";
  CO_Fr_Files[1].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[1].FileList[2].FileName="file11";
  CO_Fr_Files[1].ComponentList[1].FileList[2].FileType="inm";
```

```
    CO_Fr_Files[1].ComponentList[1].FileList[3].FileName="file11";
    CO_Fr_Files[1].ComponentList[1].FileList[3].FileType="inl";
  // Component (2,1)
    CO_Fr_Files[2].ComponentList[1].FileList[1].FileName="file21";
    CO_Fr_Files[2].ComponentList[1].FileList[1].FileType="top";
    CO_Fr_Files[2].ComponentList[1].FileList[2].FileName="file21";
    CO_Fr_Files[2].ComponentList[1].FileList[2].FileType="inm";
    CO_Fr_Files[2].ComponentList[1].FileList[3].FileName="file21";
    CO_Fr_Files[2].ComponentList[1].FileList[3].FileType="inl";
  // Component(3,1)
    CO_Fr_Files[3].ComponentList[1].FileList[1].FileName="file31";
    CO_Fr_Files[3].ComponentList[1].FileList[1].FileType="top";
    CO_Fr_Files[3].ComponentList[1].FileList[2].FileName="file31";
    CO_Fr_Files[3].ComponentList[1].FileList[2].FileType="inm";
    CO_Fr_Files[3].ComponentList[1].FileList[3].FileName="file31";
    CO_Fr_Files[3].ComponentList[1].FileList[3].FileType="inl";
    // Component (3,2)
    CO_Fr_Files[3].ComponentList[2].FileList[1].FileName="file12";
    CO_Fr_Files[3].ComponentList[2].FileList[1].FileType="top";
    CO_Fr_Files[3].ComponentList[2].FileList[2].FileName="file12";
    CO_Fr_Files[3].ComponentList[2].FileList[2].FileType="inm";
    CO_Fr_Files[3].ComponentList[2].FileList[3].FileName="file12";
    CO_Fr_Files[3].ComponentList[2].FileList[3].FileType="inl";
    //Assign Include lists
  //Component (1,1)
    // CO_Fr_Files[1][1][1].IncludeList[]={2, 3, 18, 14, -1}
    CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[1]=2;
    CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[2]=3;
    CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[3]=18;
    CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[4]=14;
    CO_Fr_Files[1].ComponentList[1].FileList[1].IncludeList[5]=-1;
    // CO_Fr_Files[1][1][2].IncludeList[]={1, 3, -1}
    CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[1]=1;
    CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[2]=3;
    CO_Fr_Files[1].ComponentList[1].FileList[2].IncludeList[3]=-1;
    // CO_Fr_Files[1][1][3].IncludeList[]={1,-1}
    CO_Fr_Files[1].ComponentList[1].FileList[3].IncludeList[1]=1;
    CO_Fr_Files[1].ComponentList[1].FileList[3].IncludeList[2]=-1;
  //Component (2,1)
    // CO_Fr_Files[2][1][1].IncludeList[]={7, 2, -1}
    CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[1]=7;
    CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[2]=2;
    CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[3]=-1;
    // CO_Fr_Files[2][1][2].IncludeList[]={8, 7, 30, -1}
    CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[1]=8;
    CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[2]=7;
    CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[3]=30;
```

```
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[4]=-1;
  // CO_Fr_Files[2][1][3].IncludeList[]={8, -1}
  CO_Fr_Files[2].ComponentList[1].FileList[3].IncludeList[1]=8;
  CO_Fr_Files[2].ComponentList[1].FileList[3].IncludeList[2]=-1;
 //Component (3,1)
  // CO_Fr_Files[3][1][1].IncludeList[]={8, 19, 11, 14, -1}
  CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[1]=8;
  CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[2]=19;
  CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[3]=11;
  CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[4]=14;
  CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[5]=-1;
  // CO_Fr_Files[3][1][2].IncludeList[]={9, 10, 11, -1}
  CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[1]=9;
  CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[2]=10;
  CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[3]=11;
  CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[4]=-1;
  // CO_Fr_Files[3][1][3].IncludeList[]={10,-1}
  CO_Fr_Files[3].ComponentList[1].FileList[3].IncludeList[1]=10;
  CO_Fr_Files[3].ComponentList[1].FileList[3].IncludeList[2]=-1;
 //Component (3,2)
  // CO_Fr_Files[3][2][1].IncludeList[]={5, 8, 9, -1}
  CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[1]=5;
  CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[2]=8;
  CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[3]=9;
  CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[4]=-1;
  // CO_Fr_Files[3][2][2].IncludeList[]={12, 10, 8, -1}
  CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[1]=12;
  CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[2]=10;
  CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[3]=8;
  CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[4]=-1;
  // CO_Fr_Files[3][2][3].IncludeList[]={7, 4, 8, -1}
  CO_Fr_Files[3].ComponentList[2].FileList[3].IncludeList[1]=7;
  CO_Fr_Files[3].ComponentList[2].FileList[3].IncludeList[2]=4;
  CO_Fr_Files[3].ComponentList[2].FileList[3].IncludeList[3]=8;
  CO_Fr_Files[3].ComponentList[2].FileList[3].IncludeList[4]=-1;
}
private static void frameworks_Structure_Example_3() {
  F=13;
  C[1]=9;   //ASX
  C[2]=2;   //CORBA
  C[3]=3;   //Collections
  C[4]=10;  //Concurrency
  C[5]=5;   //Connection
  C[6]=5;   //IPC, only those components involved in the include are considered
  C[7]=2;   //Log_Msg
  C[8]=1;   //Memory, only one component is involved in include
  C[9]=1;   //Misc, only one component has include
```

```
//C[10]=7; //Name_Service, this frameowrk does not have includes, it will be ignored
  C[10]=2; //OS, previously C[11]
  C[11]=9; //Reactor, previously C[12]
  C[12]=1; //System_V_IPC, previously 13, only one component is considered
//C[14]=2; //Timer, this does not have includes
  C[13]=1; //Token_Services, previously C[15], only one component has include
  // initialize the array structure ... new array then go over each element and initialize it.
  for (int i=1; i<=F; i++){
    CO_Fr_Files[i] = new FrameworkNode();
    CO_Fr_Files[i].FrameworkNodeNumber=i;
    for (int j=1; j<=C[i]; j++){
      CO_Fr_Files[i].ComponentList[j]= new ComponentNode();
      CO_Fr_Files[i].ComponentList[j].ID.ComponentNodeNumber=j;
      CO_Fr_Files[i].ComponentList[j].ID.ParentFrameworkNodeNumber=i;
      for (int k=1; k<=3; k++){
        CO_Fr_Files[i].ComponentList[j].FileList[k]=new FileNode();
        CO_Fr_Files[i].ComponentList[j].FileList[k].ID.FileNodeNumber=++Globals.NodeCount;
        CO_Fr_Files[i].ComponentList[j].FileList[k].ID.ParentComponentNodeNumber=j;
        CO_Fr_Files[i].ComponentList[j].FileList[k].IncludeList[1]=-1;
      }
    }
  } frameworks_Structure_Example_3_FileNames();
  frameworks_Structure_Example_3_Includes();
}
private static void frameworks_Structure_Example_3_FileNames() {
// File names Assingments
// Component(1,1)
  CO_Fr_Files[1].ComponentList[1].FileList[1].FileName="IO_Cntl_Msg";
  CO_Fr_Files[1].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[1].FileList[2].FileName="IO_Cntl_Msg";
  CO_Fr_Files[1].ComponentList[1].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[1].FileList[3].FileName="IO_Cntl_Msg";
  CO_Fr_Files[1].ComponentList[1].FileList[3].FileType="inl";
// Component(1,2)
  CO_Fr_Files[1].ComponentList[2].FileList[1].FileName="Map_Manager";
  CO_Fr_Files[1].ComponentList[2].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[2].FileList[2].FileName="Map_Manager";
  CO_Fr_Files[1].ComponentList[2].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[2].FileList[3].FileName="Map_Manager";
  CO_Fr_Files[1].ComponentList[2].FileList[3].FileType="inl";
// Component(1,3)
  CO_Fr_Files[1].ComponentList[3].FileList[1].FileName="Message_Block";
  CO_Fr_Files[1].ComponentList[3].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[3].FileList[2].FileName="Message_Block";
  CO_Fr_Files[1].ComponentList[3].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[3].FileList[3].FileName="Message_Block";
  CO_Fr_Files[1].ComponentList[3].FileList[3].FileType="inl";
```

```
// Component(1,4)
  CO_Fr_Files[1].ComponentList[4].FileList[1].FileName="Message_Queue";
  CO_Fr_Files[1].ComponentList[4].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[4].FileList[2].FileName="Message_Queue";
  CO_Fr_Files[1].ComponentList[4].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[4].FileList[3].FileName="Message_Queue";
  CO_Fr_Files[1].ComponentList[4].FileList[3].FileType="inl";
// Component(1,5)
  CO_Fr_Files[1].ComponentList[5].FileList[1].FileName="Module";
  CO_Fr_Files[1].ComponentList[5].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[5].FileList[2].FileName="Module";
  CO_Fr_Files[1].ComponentList[5].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[5].FileList[3].FileName="Module";
  CO_Fr_Files[1].ComponentList[5].FileList[3].FileType="inl";
// Component(1,6)
  CO_Fr_Files[1].ComponentList[6].FileList[1].FileName="Multiplexor";
  CO_Fr_Files[1].ComponentList[6].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[6].FileList[2].FileName="Multiplexor";
  CO_Fr_Files[1].ComponentList[6].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[6].FileList[3].FileName="Multiplexor";
  CO_Fr_Files[1].ComponentList[6].FileList[3].FileType="inl";
// Component(1,7)
  CO_Fr_Files[1].ComponentList[7].FileList[1].FileName="Stream";
  CO_Fr_Files[1].ComponentList[7].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[7].FileList[2].FileName="Stream";
  CO_Fr_Files[1].ComponentList[7].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[7].FileList[3].FileName="Stream";
  CO_Fr_Files[1].ComponentList[7].FileList[3].FileType="inl";
// Component(1,8)
  CO_Fr_Files[1].ComponentList[8].FileList[1].FileName="Stream_Modules";
  CO_Fr_Files[1].ComponentList[8].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[8].FileList[2].FileName="Stream_Modules";
  CO_Fr_Files[1].ComponentList[8].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[8].FileList[3].FileName="Stream_Modules";
  CO_Fr_Files[1].ComponentList[8].FileList[3].FileType="inl";
// Component(1,9)
  CO_Fr_Files[1].ComponentList[9].FileList[1].FileName="Task";
  CO_Fr_Files[1].ComponentList[9].FileList[1].FileType="top";
  CO_Fr_Files[1].ComponentList[9].FileList[2].FileName="Task";
  CO_Fr_Files[1].ComponentList[9].FileList[2].FileType="inm";
  CO_Fr_Files[1].ComponentList[9].FileList[3].FileName="Task";
  CO_Fr_Files[1].ComponentList[9].FileList[3].FileType="inl";
// Component (2,1)
  CO_Fr_Files[2].ComponentList[1].FileList[1].FileName="CORBA_Handler";
  CO_Fr_Files[2].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[2].ComponentList[1].FileList[2].FileName="CORBA_Handler";
  CO_Fr_Files[2].ComponentList[1].FileList[2].FileType="inm";
```

```
  CO_Fr_Files[2].ComponentList[1].FileList[3].FileName="CORBA_Handler";
  CO_Fr_Files[2].ComponentList[1].FileList[3].FileType="inl";
// Component (2,2)
  CO_Fr_Files[2].ComponentList[2].FileList[1].FileName="CORBA_Ref";
  CO_Fr_Files[2].ComponentList[2].FileList[1].FileType="top";
  CO_Fr_Files[2].ComponentList[2].FileList[2].FileName="CORBA_Ref";
  CO_Fr_Files[2].ComponentList[2].FileList[2].FileType="inm";
  CO_Fr_Files[2].ComponentList[2].FileList[3].FileName="CORBA_Ref";
  CO_Fr_Files[2].ComponentList[2].FileList[3].FileType="inl";
// Component(3,1)
  CO_Fr_Files[3].ComponentList[1].FileList[1].FileName="Set";
  CO_Fr_Files[3].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[3].ComponentList[1].FileList[2].FileName="Set";
  CO_Fr_Files[3].ComponentList[1].FileList[2].FileType="inm";
  CO_Fr_Files[3].ComponentList[1].FileList[3].FileName="Set";
  CO_Fr_Files[3].ComponentList[1].FileList[3].FileType="inl";
// Component (3,2)
  CO_Fr_Files[3].ComponentList[2].FileList[1].FileName="Stack";
  CO_Fr_Files[3].ComponentList[2].FileList[1].FileType="top";
  CO_Fr_Files[3].ComponentList[2].FileList[2].FileName="Stack";
  CO_Fr_Files[3].ComponentList[2].FileList[2].FileType="inm";
  CO_Fr_Files[3].ComponentList[2].FileList[3].FileName="Stack";
  CO_Fr_Files[3].ComponentList[2].FileList[3].FileType="inl";
// Component (3,3)
  CO_Fr_Files[3].ComponentList[3].FileList[1].FileName="SString";
  CO_Fr_Files[3].ComponentList[3].FileList[1].FileType="top";
  CO_Fr_Files[3].ComponentList[3].FileList[2].FileName="SString";
  CO_Fr_Files[3].ComponentList[3].FileList[2].FileType="inm";
  CO_Fr_Files[3].ComponentList[3].FileList[3].FileName="SString";
  CO_Fr_Files[3].ComponentList[3].FileList[3].FileType="inl";
// Component(4,1)
  CO_Fr_Files[4].ComponentList[1].FileList[1].FileName="Activation_Queue";
  CO_Fr_Files[4].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[1].FileList[2].FileName="Activation_Queue";
  CO_Fr_Files[4].ComponentList[1].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[1].FileList[3].FileName="Activation_Queue";
  CO_Fr_Files[4].ComponentList[1].FileList[3].FileType="inl";
// Component (4,2)
  CO_Fr_Files[4].ComponentList[2].FileList[1].FileName="Future";
  CO_Fr_Files[4].ComponentList[2].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[2].FileList[2].FileName="Future";
  CO_Fr_Files[4].ComponentList[2].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[2].FileList[3].FileName="Future";
  CO_Fr_Files[4].ComponentList[2].FileList[3].FileType="inl";
// Component (4,3)
  CO_Fr_Files[4].ComponentList[3].FileList[1].FileName="Method_Object";
  CO_Fr_Files[4].ComponentList[3].FileList[1].FileType="top";
```

```
  CO_Fr_Files[4].ComponentList[3].FileList[2].FileName="Method_Object";
  CO_Fr_Files[4].ComponentList[3].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[3].FileList[3].FileName="Method_Object";
  CO_Fr_Files[4].ComponentList[3].FileList[3].FileType="inl";
// Component(4,4)
  CO_Fr_Files[4].ComponentList[4].FileList[1].FileName="Process_Manager";
  CO_Fr_Files[4].ComponentList[4].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[4].FileList[2].FileName="Process_Manager";
  CO_Fr_Files[4].ComponentList[4].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[4].FileList[3].FileName="Process_Manager";
  CO_Fr_Files[4].ComponentList[4].FileList[3].FileType="inl";
// Component (4,5)
  CO_Fr_Files[4].ComponentList[5].FileList[1].FileName="Synch";
  CO_Fr_Files[4].ComponentList[5].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[5].FileList[2].FileName="Synch";
  CO_Fr_Files[4].ComponentList[5].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[5].FileList[3].FileName="Synch";
  CO_Fr_Files[4].ComponentList[5].FileList[3].FileType="inl";
// Component (4,6)
  CO_Fr_Files[4].ComponentList[6].FileList[1].FileName="Synch_Options";
  CO_Fr_Files[4].ComponentList[6].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[6].FileList[2].FileName="Synch_Options";
  CO_Fr_Files[4].ComponentList[6].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[6].FileList[3].FileName="Synch_Options";
  CO_Fr_Files[4].ComponentList[6].FileList[3].FileType="inl";
// Component (4,7)
  CO_Fr_Files[4].ComponentList[7].FileList[1].FileName="Synch_T";
  CO_Fr_Files[4].ComponentList[7].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[7].FileList[2].FileName="Synch_T";
  CO_Fr_Files[4].ComponentList[7].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[7].FileList[3].FileName="Synch_T";
  CO_Fr_Files[4].ComponentList[7].FileList[3].FileType="inl";
// Component (4,8)
  CO_Fr_Files[4].ComponentList[8].FileList[1].FileName="Thread";
  CO_Fr_Files[4].ComponentList[8].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[8].FileList[2].FileName="Thread";
  CO_Fr_Files[4].ComponentList[8].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[8].FileList[3].FileName="Thread";
  CO_Fr_Files[4].ComponentList[8].FileList[3].FileType="inl";
// Component(4,9)
  CO_Fr_Files[4].ComponentList[9].FileList[1].FileName="Thread_Manager";
  CO_Fr_Files[4].ComponentList[9].FileList[1].FileType="top";
  CO_Fr_Files[4].ComponentList[9].FileList[2].FileName="Thread_Manager";
  CO_Fr_Files[4].ComponentList[9].FileList[2].FileType="inm";
  CO_Fr_Files[4].ComponentList[9].FileList[3].FileName="Thread_Manager";
  CO_Fr_Files[4].ComponentList[9].FileList[3].FileType="inl";
// Component (4,10)
```

```
CO_Fr_Files[4].ComponentList[10].FileList[1].FileName="Token";
CO_Fr_Files[4].ComponentList[10].FileList[1].FileType="top";
CO_Fr_Files[4].ComponentList[10].FileList[2].FileName="Token";
CO_Fr_Files[4].ComponentList[10].FileList[2].FileType="inm";
CO_Fr_Files[4].ComponentList[10].FileList[3].FileName="Token";
CO_Fr_Files[4].ComponentList[10].FileList[3].FileType="inl";
// Component(5,1)
CO_Fr_Files[5].ComponentList[1].FileList[1].FileName="Acceptor";
CO_Fr_Files[5].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[5].ComponentList[1].FileList[2].FileName="Acceptor";
CO_Fr_Files[5].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[5].ComponentList[1].FileList[3].FileName="Acceptor";
CO_Fr_Files[5].ComponentList[1].FileList[3].FileType="inl";
// Component (5,2)
CO_Fr_Files[5].ComponentList[2].FileList[1].FileName="Connector";
CO_Fr_Files[5].ComponentList[2].FileList[1].FileType="top";
CO_Fr_Files[5].ComponentList[2].FileList[2].FileName="Connector";
CO_Fr_Files[5].ComponentList[2].FileList[2].FileType="inm";
CO_Fr_Files[5].ComponentList[2].FileList[3].FileName="Connector";
CO_Fr_Files[5].ComponentList[2].FileList[3].FileType="inl";
// Component (5,3)
CO_Fr_Files[5].ComponentList[3].FileList[1].FileName="Dynamic_Service";
CO_Fr_Files[5].ComponentList[3].FileList[1].FileType="top";
CO_Fr_Files[5].ComponentList[3].FileList[2].FileName="Dynamic_Service";
CO_Fr_Files[5].ComponentList[3].FileList[2].FileType="inm";
CO_Fr_Files[5].ComponentList[3].FileList[3].FileName="Dynamic_Service";
CO_Fr_Files[5].ComponentList[3].FileList[3].FileType="inl";
// Component(5,4)
CO_Fr_Files[5].ComponentList[4].FileList[1].FileName="Strategies";
CO_Fr_Files[5].ComponentList[4].FileList[1].FileType="top";
CO_Fr_Files[5].ComponentList[4].FileList[2].FileName="Strategies";
CO_Fr_Files[5].ComponentList[4].FileList[2].FileType="inm";
CO_Fr_Files[5].ComponentList[4].FileList[3].FileName="Strategies";
CO_Fr_Files[5].ComponentList[4].FileList[3].FileType="inl";
// Component (5,5)
CO_Fr_Files[5].ComponentList[5].FileList[1].FileName="Svc_Handler";
CO_Fr_Files[5].ComponentList[5].FileList[1].FileType="top";
CO_Fr_Files[5].ComponentList[5].FileList[2].FileName="Svc_Handler";
CO_Fr_Files[5].ComponentList[5].FileList[2].FileType="inm";
CO_Fr_Files[5].ComponentList[5].FileList[3].FileName="Svc_Handler";
CO_Fr_Files[5].ComponentList[5].FileList[3].FileType="inl";
// Component (6,1)
CO_Fr_Files[6].ComponentList[1].FileList[1].FileName="FIFO_Send_Msg";
CO_Fr_Files[6].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[6].ComponentList[1].FileList[2].FileName="FIFO_Send_Msg";
CO_Fr_Files[6].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[6].ComponentList[1].FileList[3].FileName="FIFO_Send_Msg";
```

```
CO_Fr_Files[6].ComponentList[1].FileList[3].FileType="inl";
// Component (6,2)
CO_Fr_Files[6].ComponentList[2].FileList[1].FileName="SOCK_Acceptor";
CO_Fr_Files[6].ComponentList[2].FileList[1].FileType="top";
CO_Fr_Files[6].ComponentList[2].FileList[2].FileName="SOCK_Acceptor";
CO_Fr_Files[6].ComponentList[2].FileList[2].FileType="inm";
CO_Fr_Files[6].ComponentList[2].FileList[3].FileName="SOCK_Acceptor";
CO_Fr_Files[6].ComponentList[2].FileList[3].FileType="inl";
// Component (6,3)
CO_Fr_Files[6].ComponentList[3].FileList[1].FileName="SOCK_Connector";
CO_Fr_Files[6].ComponentList[3].FileList[1].FileType="top";
CO_Fr_Files[6].ComponentList[3].FileList[2].FileName="SOCK_Connector";
CO_Fr_Files[6].ComponentList[3].FileList[2].FileType="inm";
CO_Fr_Files[6].ComponentList[3].FileList[3].FileName="SOCK_Connector";
CO_Fr_Files[6].ComponentList[3].FileList[3].FileType="inl";
// Component (6,4)
CO_Fr_Files[6].ComponentList[4].FileList[1].FileName="SOCK_Stream";
CO_Fr_Files[6].ComponentList[4].FileList[1].FileType="top";
CO_Fr_Files[6].ComponentList[4].FileList[2].FileName="SOCK_Stream";
CO_Fr_Files[6].ComponentList[4].FileList[2].FileType="inm";
CO_Fr_Files[6].ComponentList[4].FileList[3].FileName="SOCK_Stream";
CO_Fr_Files[6].ComponentList[4].FileList[3].FileType="inl";
// Component (6,5)
CO_Fr_Files[6].ComponentList[5].FileList[1].FileName="SPIPE_Connector";
CO_Fr_Files[6].ComponentList[5].FileList[1].FileType="top";
CO_Fr_Files[6].ComponentList[5].FileList[2].FileName="SPIPE_Connector";
CO_Fr_Files[6].ComponentList[5].FileList[2].FileType="inm";
CO_Fr_Files[6].ComponentList[5].FileList[3].FileName="SPIPE_Connector";
CO_Fr_Files[6].ComponentList[5].FileList[3].FileType="inl";
// Component (7,1)
CO_Fr_Files[7].ComponentList[1].FileList[1].FileName="Log_Msg";
CO_Fr_Files[7].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[7].ComponentList[1].FileList[2].FileName="Log_Msg";
CO_Fr_Files[7].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[7].ComponentList[1].FileList[3].FileName="Log_Msg";
CO_Fr_Files[7].ComponentList[1].FileList[3].FileType="inl";
// Component (7,2)
CO_Fr_Files[7].ComponentList[2].FileList[1].FileName="Log_Record";
CO_Fr_Files[7].ComponentList[2].FileList[1].FileType="top";
CO_Fr_Files[7].ComponentList[2].FileList[2].FileName="Log_Record";
CO_Fr_Files[7].ComponentList[2].FileList[2].FileType="inm";
CO_Fr_Files[7].ComponentList[2].FileList[3].FileName="Log_Record";
CO_Fr_Files[7].ComponentList[2].FileList[3].FileType="inl";
// Component (8,1)
CO_Fr_Files[8].ComponentList[1].FileList[1].FileName="Malloc";
CO_Fr_Files[8].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[8].ComponentList[1].FileList[2].FileName="Malloc";
```

```
CO_Fr_Files[8].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[8].ComponentList[1].FileList[3].FileName="Malloc";
CO_Fr_Files[8].ComponentList[1].FileList[3].FileType="inl";
// Component (9,1)
CO_Fr_Files[9].ComponentList[1].FileList[1].FileName="Dynamic";
CO_Fr_Files[9].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[9].ComponentList[1].FileList[2].FileName="Dynamic";
CO_Fr_Files[9].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[9].ComponentList[1].FileList[3].FileName="Dynamic";
CO_Fr_Files[9].ComponentList[1].FileList[3].FileType="inl";
// Component (10,1)
CO_Fr_Files[10].ComponentList[1].FileList[1].FileName="ACE";
CO_Fr_Files[10].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[10].ComponentList[1].FileList[2].FileName="ACE";
CO_Fr_Files[10].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[10].ComponentList[1].FileList[3].FileName="ACE";
CO_Fr_Files[10].ComponentList[1].FileList[3].FileType="inl";
// Component (10,2)
CO_Fr_Files[10].ComponentList[2].FileList[1].FileName="OS";
CO_Fr_Files[10].ComponentList[2].FileList[1].FileType="top";
CO_Fr_Files[10].ComponentList[2].FileList[2].FileName="OS";
CO_Fr_Files[10].ComponentList[2].FileList[2].FileType="inm";
CO_Fr_Files[10].ComponentList[2].FileList[3].FileName="OS";
CO_Fr_Files[10].ComponentList[2].FileList[3].FileType="inl";
// Component(11,1)
CO_Fr_Files[11].ComponentList[1].FileList[1].FileName="Event_Handler";
CO_Fr_Files[11].ComponentList[1].FileList[1].FileType="top";
CO_Fr_Files[11].ComponentList[1].FileList[2].FileName="Event_Handler";
CO_Fr_Files[11].ComponentList[1].FileList[2].FileType="inm";
CO_Fr_Files[11].ComponentList[1].FileList[3].FileName="Event_Handler";
CO_Fr_Files[11].ComponentList[1].FileList[3].FileType="inl";
// Component(11,2)
CO_Fr_Files[11].ComponentList[2].FileList[1].FileName="Event_Handler_T";
CO_Fr_Files[11].ComponentList[2].FileList[1].FileType="top";
CO_Fr_Files[11].ComponentList[2].FileList[2].FileName="Event_Handler_T";
CO_Fr_Files[11].ComponentList[2].FileList[2].FileType="inm";
CO_Fr_Files[11].ComponentList[2].FileList[3].FileName="Event_Handler_T";
CO_Fr_Files[11].ComponentList[2].FileList[3].FileType="inl";
// Component(11,3)
CO_Fr_Files[11].ComponentList[3].FileList[1].FileName="Handle_Set";
CO_Fr_Files[11].ComponentList[3].FileList[1].FileType="top";
CO_Fr_Files[11].ComponentList[3].FileList[2].FileName="Handle_Set";
CO_Fr_Files[11].ComponentList[3].FileList[2].FileType="inm";
CO_Fr_Files[11].ComponentList[3].FileList[3].FileName="Handle_Set";
CO_Fr_Files[11].ComponentList[3].FileList[3].FileType="inl";
// Component(11,4)
CO_Fr_Files[11].ComponentList[4].FileList[1].FileName="Proactor";
```

```
  CO_Fr_Files[11].ComponentList[4].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[4].FileList[2].FileName="Proactor";
  CO_Fr_Files[11].ComponentList[4].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[4].FileList[3].FileName="Proactor";
  CO_Fr_Files[11].ComponentList[4].FileList[3].FileType="inl";
// Component(11,5)
  CO_Fr_Files[11].ComponentList[5].FileList[1].FileName="Reactor";
  CO_Fr_Files[11].ComponentList[5].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[5].FileList[2].FileName="Reactor";
  CO_Fr_Files[11].ComponentList[5].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[5].FileList[3].FileName="Reactor";
  CO_Fr_Files[11].ComponentList[5].FileList[3].FileType="inl";
// Component(11,6)
  CO_Fr_Files[11].ComponentList[6].FileList[1].FileName="ReactorEx";
  CO_Fr_Files[11].ComponentList[6].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[6].FileList[2].FileName="ReactorEx";
  CO_Fr_Files[11].ComponentList[6].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[6].FileList[3].FileName="ReactorEx";
  CO_Fr_Files[11].ComponentList[6].FileList[3].FileType="inl";
// Component(11,7)
  CO_Fr_Files[11].ComponentList[7].FileList[1].FileName="Signal";
  CO_Fr_Files[11].ComponentList[7].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[7].FileList[2].FileName="Signal";
  CO_Fr_Files[11].ComponentList[7].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[7].FileList[3].FileName="Signal";
  CO_Fr_Files[11].ComponentList[7].FileList[3].FileType="inl";
// Component(11,8)
  CO_Fr_Files[11].ComponentList[8].FileList[1].FileName="Time_Value";
  CO_Fr_Files[11].ComponentList[8].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[8].FileList[2].FileName="Time_Value";
  CO_Fr_Files[11].ComponentList[8].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[8].FileList[3].FileName="Time_Value";
  CO_Fr_Files[11].ComponentList[8].FileList[3].FileType="inl";
// Component(11,9)
  CO_Fr_Files[11].ComponentList[9].FileList[1].FileName="Timer_Queue";
  CO_Fr_Files[11].ComponentList[9].FileList[1].FileType="top";
  CO_Fr_Files[11].ComponentList[9].FileList[2].FileName="Timer_Queue";
  CO_Fr_Files[11].ComponentList[9].FileList[2].FileType="inm";
  CO_Fr_Files[11].ComponentList[9].FileList[3].FileName="Timer_Queue";
  CO_Fr_Files[11].ComponentList[9].FileList[3].FileType="inl";
// Component (12,1)
  CO_Fr_Files[12].ComponentList[1].FileList[1].FileName="SV_Semaphore_Complex";
  CO_Fr_Files[12].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[12].ComponentList[1].FileList[2].FileName="SV_Semaphore_Complex";
  CO_Fr_Files[12].ComponentList[1].FileList[2].FileType="inm";
  CO_Fr_Files[12].ComponentList[1].FileList[3].FileName="SV_Semaphore_Complex";
  CO_Fr_Files[12].ComponentList[1].FileList[3].FileType="inl";
```

```
// Component (13,1)
  CO_Fr_Files[13].ComponentList[1].FileList[1].FileName="Local_Tokens";
  CO_Fr_Files[13].ComponentList[1].FileList[1].FileType="top";
  CO_Fr_Files[13].ComponentList[1].FileList[2].FileName="Local_Tokens";
  CO_Fr_Files[13].ComponentList[1].FileList[2].FileType="inm";
  CO_Fr_Files[13].ComponentList[1].FileList[3].FileName="Local_Tokens";
  CO_Fr_Files[13].ComponentList[1].FileList[3].FileType="inl";
}
private static void frameworks_Structure_Example_3_Includes() {
//Assign Include lists
//Component (1,1) No included files
//Component (1,2)
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[1]=56;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[2]=110;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[3]=155;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[4]=5;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[5]=6;
  CO_Fr_Files[1].ComponentList[2].FileList[1].IncludeList[6]=-1;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[2]=6;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[3]=4;
  CO_Fr_Files[1].ComponentList[2].FileList[2].IncludeList[4]=-1;
//Component (1,3)  CO_Fr_Files[1].ComponentList[3].FileList[1].IncludeList[1]=8;
  CO_Fr_Files[1].ComponentList[3].FileList[1].IncludeList[2]=9;
  CO_Fr_Files[1].ComponentList[3].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[1].ComponentList[3].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[1].ComponentList[3].FileList[2].IncludeList[2]=110;
  CO_Fr_Files[1].ComponentList[3].FileList[2].IncludeList[3]=9;
  CO_Fr_Files[1].ComponentList[3].FileList[2].IncludeList[4]=-1;
//Component (1,4)  CO_Fr_Files[1].ComponentList[4].FileList[1].IncludeList[1]=11;
  CO_Fr_Files[1].ComponentList[4].FileList[1].IncludeList[2]=12;
  CO_Fr_Files[1].ComponentList[4].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[1]=8;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[2]=2;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[3]=83;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[4]=12;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[5]=10;
  CO_Fr_Files[1].ComponentList[4].FileList[2].IncludeList[6]=-1;
//Component (1,5)  CO_Fr_Files[1].ComponentList[5].FileList[1].IncludeList[1]=14;
  CO_Fr_Files[1].ComponentList[5].FileList[1].IncludeList[2]=23;
  CO_Fr_Files[1].ComponentList[5].FileList[1].IncludeList[3]=15;
  CO_Fr_Files[1].ComponentList[5].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[1].ComponentList[5].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[1].ComponentList[5].FileList[2].IncludeList[2]=26;
  CO_Fr_Files[1].ComponentList[5].FileList[2].IncludeList[3]=15;
  CO_Fr_Files[1].ComponentList[5].FileList[2].IncludeList[4]=13;
  CO_Fr_Files[1].ComponentList[5].FileList[2].IncludeList[5]=-1;
```

```
//Component (1,6)  CO_Fr_Files[1].ComponentList[6].FileList[1].IncludeList[1]=17;
  CO_Fr_Files[1].ComponentList[6].FileList[1].IncludeList[2]=18;
  CO_Fr_Files[1].ComponentList[6].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[1].ComponentList[6].FileList[2].IncludeList[1]=14;
  CO_Fr_Files[1].ComponentList[6].FileList[2].IncludeList[2]=5;
  CO_Fr_Files[1].ComponentList[6].FileList[2].IncludeList[3]=18;
  CO_Fr_Files[1].ComponentList[6].FileList[2].IncludeList[5]=-1;
//Component (1,7)  CO_Fr_Files[1].ComponentList[7].FileList[1].IncludeList[1]=14;
  CO_Fr_Files[1].ComponentList[7].FileList[1].IncludeList[2]=20;
  CO_Fr_Files[1].ComponentList[7].FileList[1].IncludeList[3]=23;
  CO_Fr_Files[1].ComponentList[7].FileList[1].IncludeList[4]=21;
  CO_Fr_Files[1].ComponentList[7].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[1]=17;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[2]=116;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[3]=2;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[4]=8;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[5]=143;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[6]=14;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[7]=21;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[8]=19;
  CO_Fr_Files[1].ComponentList[7].FileList[2].IncludeList[9]=-1;
//Component (1,8)  CO_Fr_Files[1].ComponentList[8].FileList[1].IncludeList[1]=17;
  CO_Fr_Files[1].ComponentList[8].FileList[1].IncludeList[2]=18;
  CO_Fr_Files[1].ComponentList[8].FileList[1].IncludeList[3]=-1;
  // CO_Fr_Files[1][8][2].IncludeList[]={26,24,22,-1}
  CO_Fr_Files[1].ComponentList[8].FileList[2].IncludeList[1]=26;
  CO_Fr_Files[1].ComponentList[8].FileList[2].IncludeList[2]=24;
  CO_Fr_Files[1].ComponentList[8].FileList[2].IncludeList[3]=22;
  CO_Fr_Files[1].ComponentList[8].FileList[2].IncludeList[4]=-1;
//Component (1,9)  CO_Fr_Files[1].ComponentList[9].FileList[1].IncludeList[1]=26;
  CO_Fr_Files[1].ComponentList[9].FileList[1].IncludeList[2]=14;
  CO_Fr_Files[1].ComponentList[9].FileList[1].IncludeList[3]=155;
  CO_Fr_Files[1].ComponentList[9].FileList[1].IncludeList[4]=27;
  CO_Fr_Files[1].ComponentList[9].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[1].ComponentList[9].FileList[2].IncludeList[1]=159;
  CO_Fr_Files[1].ComponentList[9].FileList[2].IncludeList[2]=68;
  CO_Fr_Files[1].ComponentList[9].FileList[2].IncludeList[3]=27;
  CO_Fr_Files[1].ComponentList[9].FileList[2].IncludeList[4]=-1;
//Component (2,1)  CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[1]=29;
  CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[2]=30;
  CO_Fr_Files[2].ComponentList[1].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[1]=155;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[2]=157;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[3]=158;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[4]=30;
  CO_Fr_Files[2].ComponentList[1].FileList[2].IncludeList[5]=-1;
//Component (2,2)  CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[1]=32;
```

```
CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[2]=33;
CO_Fr_Files[2].ComponentList[2].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[1]=33;
CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[2]=31;
CO_Fr_Files[2].ComponentList[2].FileList[2].IncludeList[3]=-1;
//Component (3,1) CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[1]=35;
CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[2]=36;
CO_Fr_Files[3].ComponentList[1].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[1]=116;
CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[2]=36;
CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[3]=34;
CO_Fr_Files[3].ComponentList[1].FileList[2].IncludeList[4]=-1;
//Component (3,2) CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[1]=38;
CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[2]=39;
CO_Fr_Files[3].ComponentList[2].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[1]=116;
CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[2]=39;
CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[3]=37;
CO_Fr_Files[3].ComponentList[2].FileList[2].IncludeList[4]=-1;
//Component (3,3) CO_Fr_Files[3].ComponentList[3].FileList[1].IncludeList[1]=110;
CO_Fr_Files[3].ComponentList[3].FileList[1].IncludeList[2]=155;
CO_Fr_Files[3].ComponentList[3].FileList[1].IncludeList[3]=41;
CO_Fr_Files[3].ComponentList[3].FileList[1].IncludeList[4]=42;
CO_Fr_Files[3].ComponentList[3].FileList[1].IncludeList[5]=-1;
CO_Fr_Files[3].ComponentList[3].FileList[2].IncludeList[1]=116;
CO_Fr_Files[3].ComponentList[3].FileList[2].IncludeList[2]=42;
CO_Fr_Files[3].ComponentList[3].FileList[2].IncludeList[3]=-1;
//Component (4,1) CO_Fr_Files[4].ComponentList[1].FileList[1].IncludeList[1]=44;
CO_Fr_Files[4].ComponentList[1].FileList[1].IncludeList[2]=-1;
CO_Fr_Files[4].ComponentList[1].FileList[2].IncludeList[1]=56;
CO_Fr_Files[4].ComponentList[1].FileList[2].IncludeList[2]=11;
CO_Fr_Files[4].ComponentList[1].FileList[2].IncludeList[3]=50;
CO_Fr_Files[4].ComponentList[1].FileList[2].IncludeList[4]=-1;
//Component (4,2) CO_Fr_Files[4].ComponentList[2].FileList[1].IncludeList[1]=47;
CO_Fr_Files[4].ComponentList[2].FileList[1].IncludeList[2]=-1;
CO_Fr_Files[4].ComponentList[2].FileList[2].IncludeList[1]=56;
CO_Fr_Files[4].ComponentList[2].FileList[2].IncludeList[2]=46;
CO_Fr_Files[4].ComponentList[2].FileList[2].IncludeList[3]=-1;
//Component (4,3) CO_Fr_Files[4].ComponentList[3].FileList[1].IncludeList[1]=50;
CO_Fr_Files[4].ComponentList[3].FileList[1].IncludeList[2]=-1;
CO_Fr_Files[4].ComponentList[3].FileList[2].IncludeList[1]=119;
CO_Fr_Files[4].ComponentList[3].FileList[2].IncludeList[2]=-1;
//Component (4,4) CO_Fr_Files[4].ComponentList[4].FileList[1].IncludeList[1]=53;
CO_Fr_Files[4].ComponentList[4].FileList[1].IncludeList[2]=54;
CO_Fr_Files[4].ComponentList[4].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[4].ComponentList[4].FileList[2].IncludeList[1]=56;
CO_Fr_Files[4].ComponentList[4].FileList[2].IncludeList[2]=54;
```

```
  CO_Fr_Files[4].ComponentList[4].FileList[2].IncludeList[3]=-1;
//Component (4,5)  CO_Fr_Files[4].ComponentList[5].FileList[1].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[5].FileList[1].IncludeList[2]=56;
  CO_Fr_Files[4].ComponentList[5].FileList[1].IncludeList[3]=143;
  CO_Fr_Files[4].ComponentList[5].FileList[1].IncludeList[4]=57;
  CO_Fr_Files[4].ComponentList[5].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[4].ComponentList[5].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[4].ComponentList[5].FileList[2].IncludeList[2]=149;
  CO_Fr_Files[4].ComponentList[5].FileList[2].IncludeList[3]=57;
  CO_Fr_Files[4].ComponentList[5].FileList[2].IncludeList[4]=62;
  CO_Fr_Files[4].ComponentList[5].FileList[2].IncludeList[5]=-1;
//Component (4,6)  CO_Fr_Files[4].ComponentList[6].FileList[1].IncludeList[1]=59;
  CO_Fr_Files[4].ComponentList[6].FileList[1].IncludeList[2]=116;
  CO_Fr_Files[4].ComponentList[6].FileList[1].IncludeList[3]=-1;
//Component (4,7)  CO_Fr_Files[4].ComponentList[7].FileList[1].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[7].FileList[1].IncludeList[2]=143;
  CO_Fr_Files[4].ComponentList[7].FileList[1].IncludeList[3]=62;
  CO_Fr_Files[4].ComponentList[7].FileList[1].IncludeList[4]=63;
  CO_Fr_Files[4].ComponentList[7].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[4].ComponentList[7].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[4].ComponentList[7].FileList[2].IncludeList[2]=56;
  CO_Fr_Files[4].ComponentList[7].FileList[2].IncludeList[3]=63;
  CO_Fr_Files[4].ComponentList[7].FileList[2].IncludeList[4]=61;
  CO_Fr_Files[4].ComponentList[7].FileList[2].IncludeList[5]=-1;
  CO_Fr_Files[4].ComponentList[7].FileList[3].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[7].FileList[3].IncludeList[2]=-1;
//Component (4,8)  CO_Fr_Files[4].ComponentList[8].FileList[1].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[8].FileList[1].IncludeList[2]=66;
  CO_Fr_Files[4].ComponentList[8].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[4].ComponentList[8].FileList[2].IncludeList[1]=116;
  CO_Fr_Files[4].ComponentList[8].FileList[2].IncludeList[2]=66;
  CO_Fr_Files[4].ComponentList[8].FileList[2].IncludeList[3]=-1;
//Component (4,9)  CO_Fr_Files[4].ComponentList[9].FileList[1].IncludeList[1]=68;
  CO_Fr_Files[4].ComponentList[9].FileList[1].IncludeList[2]=69;
  CO_Fr_Files[4].ComponentList[9].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[4].ComponentList[9].FileList[2].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[9].FileList[2].IncludeList[2]=56;
  CO_Fr_Files[4].ComponentList[9].FileList[2].IncludeList[3]=69;
  CO_Fr_Files[4].ComponentList[9].FileList[2].IncludeList[4]=-1;
//Component (4,10)  CO_Fr_Files[4].ComponentList[10].FileList[1].IncludeList[1]=65;
  CO_Fr_Files[4].ComponentList[10].FileList[1].IncludeList[2]=143;
  CO_Fr_Files[4].ComponentList[10].FileList[1].IncludeList[3]=71;
  CO_Fr_Files[4].ComponentList[10].FileList[1].IncludeList[4]=72;
  CO_Fr_Files[4].ComponentList[10].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[4].ComponentList[10].FileList[2].IncludeList[1]=56;
  CO_Fr_Files[4].ComponentList[10].FileList[2].IncludeList[2]=72;
  CO_Fr_Files[4].ComponentList[10].FileList[2].IncludeList[3]=-1;
```

```
//Component (5,1) CO_Fr_Files[5].ComponentList[1].FileList[1].IncludeList[1]=116;
  CO_Fr_Files[5].ComponentList[1].FileList[1].IncludeList[2]=74;
  CO_Fr_Files[5].ComponentList[1].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[1]=155;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[2]=159;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[3]=86;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[4]=83;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[5]=75;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[6]=73;
  CO_Fr_Files[5].ComponentList[1].FileList[2].IncludeList[7]=-1;
//Component (5,2) CO_Fr_Files[5].ComponentList[2].FileList[1].IncludeList[1]=77;
  CO_Fr_Files[5].ComponentList[2].FileList[1].IncludeList[2]=-1;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[1]=155;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[2]=159;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[3]=146;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[4]=5;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[5]=86;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[6]=83;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[7]=78;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[8]=76;
  CO_Fr_Files[5].ComponentList[2].FileList[2].IncludeList[9]=-1;
//Component (5,3) CO_Fr_Files[5].ComponentList[3].FileList[1].IncludeList[1]=155;
  CO_Fr_Files[5].ComponentList[3].FileList[1].IncludeList[2]=160;
  CO_Fr_Files[5].ComponentList[3].FileList[1].IncludeList[3]=80;
  CO_Fr_Files[5].ComponentList[3].FileList[1].IncludeList[4]=11;
  CO_Fr_Files[5].ComponentList[3].FileList[2].IncludeList[1]=79;
  CO_Fr_Files[5].ComponentList[3].FileList[2].IncludeList[2]=-1;
//Component (5,4) CO_Fr_Files[5].ComponentList[4].FileList[1].IncludeList[1]=134;
  CO_Fr_Files[5].ComponentList[4].FileList[1].IncludeList[2]=137;
  CO_Fr_Files[5].ComponentList[4].FileList[1].IncludeList[3]=83;
  CO_Fr_Files[5].ComponentList[4].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[5].ComponentList[4].FileList[2].IncludeList[1]=122;
  CO_Fr_Files[5].ComponentList[4].FileList[2].IncludeList[2]=-1;
//Component (5,5) CO_Fr_Files[5].ComponentList[5].FileList[1].IncludeList[1]=86;
  CO_Fr_Files[5].ComponentList[5].FileList[1].IncludeList[2]=113;
  CO_Fr_Files[5].ComponentList[5].FileList[1].IncludeList[3]=87;
  CO_Fr_Files[5].ComponentList[5].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[1]=59;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[2]=26;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[3]=155;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[4]=62;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[5]=87;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[6]=85;
  CO_Fr_Files[5].ComponentList[5].FileList[2].IncludeList[7]=-1;
//Component (7,1) CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[1]=154;
  CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[2]=116;
  CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[3]=117;
```

```
CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[4]=65;
CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[5]=56;
CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[6]=101;
CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[7]=89;
CO_Fr_Files[7].ComponentList[1].FileList[1].IncludeList[8]=-1;
CO_Fr_Files[7].ComponentList[1].FileList[2].IncludeList[1]=107;
CO_Fr_Files[7].ComponentList[1].FileList[2].IncludeList[2]=105;
CO_Fr_Files[7].ComponentList[1].FileList[2].IncludeList[3]=-1;
//Component (7,2) CO_Fr_Files[7].ComponentList[2].FileList[1].IncludeList[1]=107;
CO_Fr_Files[7].ComponentList[2].FileList[1].IncludeList[2]=-1;
CO_Fr_Files[7].ComponentList[2].FileList[2].IncludeList[1]=116;
CO_Fr_Files[7].ComponentList[2].FileList[2].IncludeList[2]=108;
CO_Fr_Files[7].ComponentList[2].FileList[2].IncludeList[3]=-1;
// Component (11,1) CO_Fr_Files[11].ComponentList[1].FileList[1].IncludeList[1]=122;
CO_Fr_Files[11].ComponentList[1].FileList[1].IncludeList[2]=8;
CO_Fr_Files[11].ComponentList[1].FileList[1].IncludeList[3]=123;
CO_Fr_Files[11].ComponentList[1].FileList[1].IncludeList[4]=-1;
CO_Fr_Files[11].ComponentList[1].FileList[2].IncludeList[1]=116;
CO_Fr_Files[11].ComponentList[1].FileList[2].IncludeList[2]=123;
CO_Fr_Files[11].ComponentList[1].FileList[2].IncludeList[3]=-1;
CO_Fr_Files[11].ComponentList[1].FileList[3].IncludeList[1]=122;
CO_Fr_Files[11].ComponentList[1].FileList[3].IncludeList[2]=-1;
// Component (11,2) CO_Fr_Files[11].ComponentList[2].FileList[1].IncludeList[1]=125;
CO_Fr_Files[11].ComponentList[2].FileList[1].IncludeList[2]=126;
CO_Fr_Files[11].ComponentList[2].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[11].ComponentList[2].FileList[2].IncludeList[1]=122;
CO_Fr_Files[11].ComponentList[2].FileList[2].IncludeList[2]=126;
CO_Fr_Files[11].ComponentList[2].FileList[2].IncludeList[3]=-1;
// Component (11,3) CO_Fr_Files[11].ComponentList[3].FileList[1].IncludeList[1]=128;
CO_Fr_Files[11].ComponentList[3].FileList[1].IncludeList[2]=129;
CO_Fr_Files[11].ComponentList[3].FileList[1].IncludeList[3]=-1;
// CO_Fr_Files[11][3][2].IncludeList[]={293,306,-1}
CO_Fr_Files[11].ComponentList[3].FileList[2].IncludeList[1]=116;
CO_Fr_Files[11].ComponentList[3].FileList[2].IncludeList[2]=129;
CO_Fr_Files[11].ComponentList[3].FileList[2].IncludeList[3]=-1;
// Component (11,4) CO_Fr_Files[11].ComponentList[4].FileList[1].IncludeList[1]=131;
CO_Fr_Files[11].ComponentList[4].FileList[1].IncludeList[2]=132;
CO_Fr_Files[11].ComponentList[4].FileList[1].IncludeList[3]=-1;
CO_Fr_Files[11].ComponentList[4].FileList[2].IncludeList[1]=119;
CO_Fr_Files[11].ComponentList[4].FileList[2].IncludeList[2]=146;
CO_Fr_Files[11].ComponentList[4].FileList[2].IncludeList[3]=122;
CO_Fr_Files[11].ComponentList[4].FileList[2].IncludeList[4]=132;
CO_Fr_Files[11].ComponentList[4].FileList[2].IncludeList[5]=-1;
// Component (11,5) CO_Fr_Files[11].ComponentList[5].FileList[1].IncludeList[1]=62;
CO_Fr_Files[11].ComponentList[5].FileList[1].IncludeList[2]=92;
CO_Fr_Files[11].ComponentList[5].FileList[1].IncludeList[3]=95;
CO_Fr_Files[11].ComponentList[5].FileList[1].IncludeList[4]=134;
```

```
  CO_Fr_Files[11].ComponentList[5].FileList[1].IncludeList[5]=-1;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[1]=128;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[2]=146;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[3]=140;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[4]=65;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[5]=71;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[6]=156;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[7]=98;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[8]=152;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[9]=135;
  CO_Fr_Files[11].ComponentList[5].FileList[2].IncludeList[10]=-1;
// Component (11,6)  CO_Fr_Files[11].ComponentList[6].FileList[1].IncludeList[1]=137;
  CO_Fr_Files[11].ComponentList[6].FileList[1].IncludeList[2]=138;
  CO_Fr_Files[11].ComponentList[6].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[1]=143;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[2]=146;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[3]=122;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[4]=11;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[5]=71;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[6]=56;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[7]=152;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[8]=138;
  CO_Fr_Files[11].ComponentList[6].FileList[2].IncludeList[9]=-1;
// Component (11,7)  CO_Fr_Files[11].ComponentList[7].FileList[1].IncludeList[1]=140;
  CO_Fr_Files[11].ComponentList[7].FileList[1].IncludeList[2]=141;
  CO_Fr_Files[11].ComponentList[7].FileList[1].IncludeList[3]=-1;
  CO_Fr_Files[11].ComponentList[7].FileList[2].IncludeList[1]=56;
  CO_Fr_Files[11].ComponentList[7].FileList[2].IncludeList[2]=122;
  CO_Fr_Files[11].ComponentList[7].FileList[2].IncludeList[3]=35;
  CO_Fr_Files[11].ComponentList[7].FileList[2].IncludeList[4]=141;
  CO_Fr_Files[11].ComponentList[7].FileList[2].IncludeList[5]=-1;
// Component (11,8)  CO_Fr_Files[11].ComponentList[8].FileList[1].IncludeList[1]=116;
  CO_Fr_Files[11].ComponentList[8].FileList[1].IncludeList[2]=143;
  CO_Fr_Files[11].ComponentList[8].FileList[1].IncludeList[3]=144;
  CO_Fr_Files[11].ComponentList[8].FileList[1].IncludeList[4]=-1;
  CO_Fr_Files[11].ComponentList[8].FileList[2].IncludeList[1]=117;
  CO_Fr_Files[11].ComponentList[8].FileList[2].IncludeList[2]=144;
  CO_Fr_Files[11].ComponentList[8].FileList[2].IncludeList[3]=56;
  CO_Fr_Files[11].ComponentList[8].FileList[2].IncludeList[4]=147;
  CO_Fr_Files[11].ComponentList[8].FileList[2].IncludeList[5]=-1;
// Component (11,9)  CO_Fr_Files[11].ComponentList[9].FileList[1].IncludeList[1]=146;
  CO_Fr_Files[11].ComponentList[9].FileList[1].IncludeList[2]=147;
  CO_Fr_Files[11].ComponentList[9].FileList[1].IncludeList[3]=-1;
}
public static void initializeFor_Example_1() {
  frameworks_Structure_Example_1();
```

```
    sharedPools_Structure_Example_1();
  }
  public static void initializeFor_Example_1_OLD() {
    frameworks_Structure_Example_1();
    sharedPools_Structure_Example_1();
  }
  public static void initializeFor_Example_2() {
    frameworks_Structure_Example_2();
    sharedPools_Structure_Example_2();
  }
  public static void initializeFor_Example_3() {
    frameworks_Structure_Example_3();
    sharedPools_Structure_Example_3();
  }
  public static void reportFileStructure(String output){
    System.out.println("\n"+output+"\n");
    return;
  }
  private static void sharedPools_Structure_Example_1() {
    S=2;
    SF[1]=2;
    SF[2]=1;
// initialize the array structure ... new array then go over each element and initialize it.
    for (int i=1;i<=S;i++){
      CO_Sh_Files[i]=new SharedPoolNode();
      CO_Sh_Files[i].SharedPoolNodeNumber=i;
      for (int j=1;j<=SF[i];j++){
        CO_Sh_Files[i].SharedFileList[j]=new SharedFileNode();
        CO_Sh_Files[i].SharedFileList[j].ID.ParentSharedPoolNodeNumber=i;
        CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber=++Globals.NodeCount;
      }
    }
//Assignments — Permitted frameworks
// SharedPool(1)
   CO_Sh_Files[1].PermittedFrameworks[1]=1;
   CO_Sh_Files[1].PermittedFrameworks[2]=2;
   CO_Sh_Files[1].PermittedFrameworks[3]=-1;
// SharedPool(2)
   CO_Sh_Files[2].PermittedFrameworks[1]=1;
   CO_Sh_Files[2].PermittedFrameworks[2]=-1;
//Assignments —— File names
// SharedPool(1)
   CO_Sh_Files[1].SharedFileList[1].FileName="sh_file11";
   CO_Sh_Files[1].SharedFileList[1].FileType="s";
   CO_Sh_Files[1].SharedFileList[2].FileName="sh_file12";
   CO_Sh_Files[1].SharedFileList[2].FileType="s";
```

```
// SharedPool(2)
  CO_Sh_Files[2].SharedFileList[1].FileName="sh_file21";
  CO_Sh_Files[2].SharedFileList[1].FileType="s";
//Assign Include lists
//SharedPool(1)
  CO_Sh_Files[1].SharedFileList[1].IncludeList[1]=14;
  CO_Sh_Files[1].SharedFileList[1].IncludeList[2]=-1;
  CO_Sh_Files[1].SharedFileList[2].IncludeList[1]=-1;
//SharedPools(2)
  CO_Sh_Files[2].SharedFileList[1].IncludeList[1]=-1;
}
private static void sharedPools_Structure_Example_1_OLD() {
  S=2;
  SF[1]=2;
  SF[2]=1;
// initialize the array structure ... new array then go over each element and initialize it.
  for (int i=1;i<=S;i++){
    CO_Sh_Files[i]=new SharedPoolNode();
    CO_Sh_Files[i].SharedPoolNodeNumber=i;
    for (int j=1;j<=SF[i];j++){
      CO_Sh_Files[i].SharedFileList[j]=new SharedFileNode();
      CO_Sh_Files[i].SharedFileList[j].ID.ParentSharedPoolNodeNumber=i;
      CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber=++Globals.NodeCount;
    }
  }
//Assignments — Permitted frameworks
// SharedPool(1)
  CO_Sh_Files[1].PermittedFrameworks[1]=1;
  CO_Sh_Files[1].PermittedFrameworks[2]=2;
  CO_Sh_Files[1].PermittedFrameworks[3]=-1;
// SharedPool(2)
  CO_Sh_Files[2].PermittedFrameworks[1]=1;
  CO_Sh_Files[2].PermittedFrameworks[2]=-1;
//Assignments —— File names
// SharedPool(1)
  CO_Sh_Files[1].SharedFileList[1].FileName="sh_file11";
  CO_Sh_Files[1].SharedFileList[1].FileType="s";
  CO_Sh_Files[1].SharedFileList[2].FileName="sh_file12";
  CO_Sh_Files[1].SharedFileList[2].FileType="s";
// SharedPool(2)
  CO_Sh_Files[2].SharedFileList[1].FileName="sh_file21";
  CO_Sh_Files[2].SharedFileList[1].FileType="s";
//Assign Include lists
//SharedPool(1)
  CO_Sh_Files[1].SharedFileList[1].IncludeList[1]=11;
  CO_Sh_Files[1].SharedFileList[1].IncludeList[2]=16;
  CO_Sh_Files[1].SharedFileList[1].IncludeList[3]=-1;
```

```
    CO_Sh_Files[1].SharedFileList[2].IncludeList[1]=14;
    CO_Sh_Files[1].SharedFileList[2].IncludeList[2]=16;
    CO_Sh_Files[1].SharedFileList[2].IncludeList[3]=-1;
  //SharedPools(2)
    CO_Sh_Files[2].SharedFileList[1].IncludeList[1]=13;
    CO_Sh_Files[2].SharedFileList[1].IncludeList[2]=14;
    CO_Sh_Files[2].SharedFileList[1].IncludeList[4]=-1;
  }
  private static void sharedPools_Structure_Example_2() {
    S=2;
    SF[1]=2;
    SF[2]=1;
  // initialize the array structure ... new array then go over each element and initialize it.
    for (int i=1;i<=S;i++){
      CO_Sh_Files[i]=new SharedPoolNode();
      CO_Sh_Files[i].SharedPoolNodeNumber=i;
      for (int j=1;j<=SF[i];j++){
        CO_Sh_Files[i].SharedFileList[j]=new SharedFileNode();
        CO_Sh_Files[i].SharedFileList[j].ID.ParentSharedPoolNodeNumber=i;
        CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber=++Globals.NodeCount;
      }
    }
  //Assignments — Permitted frameworks
  // SharedPool(1)
    CO_Sh_Files[1].PermittedFrameworks[1]=1;
    CO_Sh_Files[1].PermittedFrameworks[2]=2;
    CO_Sh_Files[1].PermittedFrameworks[3]=-1;
  // SharedPool(2)
    CO_Sh_Files[2].PermittedFrameworks[1]=2;
    CO_Sh_Files[2].PermittedFrameworks[2]=3;
    CO_Sh_Files[2].PermittedFrameworks[3]=-1;
  //Assignments —— File names
  // SharedPool(1)
    CO_Sh_Files[1].SharedFileList[1].FileName="sh_file11";
    CO_Sh_Files[1].SharedFileList[1].FileType="s";
    CO_Sh_Files[1].SharedFileList[2].FileName="sh_file12";
    CO_Sh_Files[1].SharedFileList[2].FileType="s";
  // SharedPool(2)
    CO_Sh_Files[2].SharedFileList[1].FileName="sh_file21";
    CO_Sh_Files[2].SharedFileList[1].FileType="s";
  //Assign Include lists
  //SharedPool(1)
    CO_Sh_Files[1].SharedFileList[1].IncludeList[1]=11;
    CO_Sh_Files[1].SharedFileList[1].IncludeList[2]=16;
    CO_Sh_Files[1].SharedFileList[1].IncludeList[3]=-1;
    CO_Sh_Files[1].SharedFileList[2].IncludeList[1]=14;
    CO_Sh_Files[1].SharedFileList[2].IncludeList[2]=16;
```

```
    CO_Sh_Files[1].SharedFileList[2].IncludeList[3]=-1;
  //SharedPools(2)
    CO_Sh_Files[2].SharedFileList[1].IncludeList[1]=13;
    CO_Sh_Files[2].SharedFileList[1].IncludeList[2]=14;
    CO_Sh_Files[2].SharedFileList[1].IncludeList[4]=-1;
  }
  private static void sharedPools_Structure_Example_3() {
    S=1;
    SF[1]=7;
  // initialize the array structure ... new array then go over each element and initialize it.
    for (int i=1;i<=S;i++){
      CO_Sh_Files[i]=new SharedPoolNode();
      CO_Sh_Files[i].SharedPoolNodeNumber=i;
      for (int j=1;j<=SF[i];j++){
        CO_Sh_Files[i].SharedFileList[j]=new SharedFileNode();
        CO_Sh_Files[i].SharedFileList[j].ID.ParentSharedPoolNodeNumber=i;
        CO_Sh_Files[i].SharedFileList[j].ID.FileNodeNumber=++Globals.NodeCount;
        CO_Sh_Files[i].SharedFileList[j].IncludeList[1]=-1;
      }
    }
  //Assignments — Permitted frameworks
  // SharedPool(1)
    CO_Sh_Files[1].PermittedFrameworks[1]=1;
    CO_Sh_Files[1].PermittedFrameworks[2]=2;
    CO_Sh_Files[1].PermittedFrameworks[3]=10;
    CO_Sh_Files[1].PermittedFrameworks[4]=-1;
  //Assignments —— File names
  // SharedPool(1)
    CO_Sh_Files[1].SharedFileList[1].FileName="config";
    CO_Sh_Files[1].SharedFileList[1].FileType="s";
    CO_Sh_Files[1].SharedFileList[2].FileName="Service_Config";
    CO_Sh_Files[1].SharedFileList[2].FileType="s";
    CO_Sh_Files[1].SharedFileList[3].FileName="Pipe";
    CO_Sh_Files[1].SharedFileList[3].FileType="s";
    CO_Sh_Files[1].SharedFileList[4].FileName="CORBA";
    CO_Sh_Files[1].SharedFileList[4].FileType="s";
    CO_Sh_Files[1].SharedFileList[5].FileName="daemon";
    CO_Sh_Files[1].SharedFileList[5].FileType="s";
    CO_Sh_Files[1].SharedFileList[6].FileName="Service_Object";
    CO_Sh_Files[1].SharedFileList[6].FileType="s";
    CO_Sh_Files[1].SharedFileList[7].FileName="Service_Repository";
    CO_Sh_Files[1].SharedFileList[7].FileType="s";
  //Assign Include lists
   //SharedPool(1)
  // None
  }
  }
```

# Appendix A2

# A Prototype For CO–Toolkits' File Structures: Example#1 – An Artificial Example

## A2.1 Overview

This Appendix presents the first artificial example that has been used to demonstrate our prototype for CO–Toolkits' file structures. Section A2.3 shows that this example passes the prototype test (i.e. each predicate returns STATUS=true).

## A2.2 Example#1: File Structure

FILE STRUCTURE: As Extracted From The Current Example

```
Framework(1)
    Component(1,1)
        FileNode(1):   file11.top   Includes FileNodes:   2  13  15
        FileNode(2):   file11.inm   Includes FileNodes:   3  5
        FileNode(3):   file11.inl   Includes FileNodes:
    Component(1,2)
        FileNode(4):   file12.top   Includes FileNodes:   5
        FileNode(5):   file12.inm   Includes FileNodes:   6
        FileNode(6):   file12.inl   Includes FileNodes:
Framework(2)
    Component(2,1)
        FileNode(7):   file21.top   Includes FileNodes:   8
        FileNode(8):   file21.inm   Includes FileNodes:   9
        FileNode(9):   file21.inl   Includes FileNodes:
    Component(2,2)
        FileNode(10):  file22.top   Includes FileNodes:   11  13  14
        FileNode(11):  file22.inm   Includes FileNodes:   12  8
        FileNode(12):  file22.inl   Includes FileNodes:
```

SharedPool(1)
    FileNode(13): sh_file11.s Includes FileNodes: 14
    FileNode(14): sh_file12.s Includes FileNodes:
  Permitted Frameworks to this SharedPool are: Framework(1) Framework(2)
SharedPool(2)
    FileNode(15): sh_file21.s Includes FileNodes:
  Permitted Frameworks to this SharedPool are: Framework(1)

FILE STRUCTURE display is DONE

# A2.3 Example#1: Output Report

Checking for GoodFileStructure_predicate4_5: GoodFileStructure = predicates 4_6, 4_7, and 4_8

Checking for Predicate 4_6: OnlyFourFileCategories

Checking for Predicate 4_6 is complete —— STATUS: true

Checking for Predicate 4_7: GoodDistinctionOfFiles= Predicates 4_9, and 4_10

Checking for Predicate 4_9: Fr_SetsOfFilesShouldBeDistinct

Checking for Predicate 4_9 is complete —— STATUS: true

Checking for Predicate 4_10: ComponentsMainFilesShouldBeDistinct
Checking for Predicate 4_10 is complete —— STATUS: true

Checking for Predicate 4_7 is complete —— STATUS: true

Checking for Predicate 4_8: GoodFileInclusionRelation = predicates 4_11, 4_12, and 4_13

Checking for Predicate 4_11: GoodUseOfFiles = predicates 4_14, 4_15, 4_16, and 4_17

Checking for Predicate 4_12: InclusionLoopsMayNotExist

Checking for Predicate 4_12 is complete —— STATUS: true

Checking for Predicate 4_13: GoodUseOf_top_Files = predicates 4_18, 4_19, and 4_20

Checking for predicate 4_17: NoFileMayIncludeA_top_File

Checking for predicate 4_17 is complete —— STATUS: true

Checking for predicate 4_18: FromSame_Fr_A_top_FileMayIncludeOnlyIts_intermediate_File

Checking for predicate 4_18 is complete —— STATUS: true

Checking for predicate 4_19: A_top_FileMayInclude_shared_FilesFromPermittedSharedPools

Checking for predicate 4_19 is complete —— STATUS: true

Checking for Predicate 4_13 is complete — STATUS: true

Checking for Predicate 4_14: GoodUseOf_intermediate_Files = predicates 4_21 and 4_22

Checking for predicate 4_20: An_intermediate_fileMayIncludeOnlyIts_internal_FileAndMayIncludeOther_intermediate_FilesFromSame_Fr

Checking for predicate 4_20 is complete —— STATUS: true

Checking for Predicate 4_14 is complete —— STATUS: true

Checking for Predicate 4_15: GoodUseOf_internal_Files = predicates 4_23 and 4_24

Checking for predicate 4_21: An_internal_FileMayNotIncludeAnyFile

Checking for predicate 4_21 is complete — STATUS:true

Checking for predicate 4_22: NoFileMayIncludeAn_Internal_FileExceptIts_intermediate_File

Checking for predicate 4_22 is complete —— STATUS:true

Checking for Predicate 4_15 is complete —— STATUS: true

Checking for Predicate 4_11 is complete —— STATUS: true

Checking for Predicate 4_12: InclusionLoopsMayNotExist

Checking for Predicate 4_12 is complete —— STATUS: true

Checking for Predicate 4_8 is complete —— STATUS: true

Checking for GoodFileStructure_predicate4_5 is complete —— STATUS:true

# Appendix A3

# VA Prototype For CO–Toolkits' File Structures: Example#2 – An Artificial Example

## A3.1 Overview

This Appendix presents the second artificial example that has been used to demonstrate our prototype for CO–Toolkits' file structures. Section A3.3 presents the rules violations reported by this prototype for this example.

## A3.2 Example#2: File Structure

```
FILE STRUCTURE: As Extracted From The Current Example

Framework(1)
    Component(1,1)
        FileNode(1):   file11.top  Includes FileNodes:  2   3   18  14
        FileNode(2):   file11.inm  Includes FileNodes:  1   3
        FileNode(3):   file11.inl  Includes FileNodes:  1
Framework(2)
    Component(2,1)
        FileNode(4):   file21.top  Includes FileNodes:  7   2
        FileNode(5):   file21.inm  Includes FileNodes:  8   7   30
        FileNode(6):   file21.inl  Includes FileNodes:  8
Framework(3)
    Component(3,1)
        FileNode(7):   file31.top  Includes FileNodes:  8   19  11  14
        FileNode(8):   file31.inm  Includes FileNodes:  9   10  11
        FileNode(9):   file31.inl  Includes FileNodes:  10
    Component(3,2)
        FileNode(10):  file12.top  Includes FileNodes:  5   8   9
        FileNode(11):  file12.inm  Includes FileNodes:  12  10  8
        FileNode(12):  file12.inl  Includes FileNodes:  7   4   8

SharedPool(1)
        FileNode(13):  sh_file11.s      Includes FileNodes:  11  16
        FileNode(14):  sh_file12.s      Includes FileNodes:  14  16
    Permitted Frameworks to this SharedPool are:    Framework(1)  Framework(2)
SharedPool(2)
        FileNode(15):  sh_file21.s      Includes FileNodes:  13  14
    Permitted Frameworks to this SharedPool are:    Framework(2)  Framework(3)

FILE STRUCTURE display is DONE
```

# A3.3 Example#2: Output Report

```
Checking for GoodFileStructure_predicate4_5: GoodFileStructure = predicates 4_6, 4_7, and 4_8

Checking for Predicate 4_6: OnlyFourFileCategories

Checking for Predicate 4_6 is complete —— STATUS: true

==========================================================

Checking for Predicate 4_7: GoodDistinctionOfFiles= Predicates 4_9, and 4_10

Checking for Predicate 4_9: Fr_SetsOfFilesShouldBeDistinct

Checking for Predicate 4_9 is complete —— STATUS: true

==========================================================

Checking for Predicate 4_10: ComponentsMainFilesShouldBeDistinct
Checking for Predicate 4_10 is complete —— STATUS: true
==========================================================

Checking for Predicate 4_7 is complete —— STATUS: true
==========================================================

Checking for Predicate 4_8: GoodFileInclusionRelation = predicates 4_11, 4_12, and 4_13

Checking for Predicate 4_11: GoodUseOfFiles = predicates 4_14, 4_15, 4_16, and 4_17

Checking for Predicate 4_12: InclusionLoopsMayNotExist
ERROR(Predicate4_13): File file11.top with FileNode(1) from Framework(1) and Component(1,1) includes FileNode(18) that has no existing file
ERROR(Predicate4_13): File file21.inm with FileNode(5) from Framework(2) and Component(2,1) includes FileNode(30) that has no existing file
ERROR(Predicate4_13): File file31.top with FileNode(7) from Framework(3) and Component(3,1) includes FileNode(19) that has no existing file
ERROR(Predicate4_13): SharedFile sh_file11.s with FileNode(13) from SharedPool(1) includes FileNode(16) that has no existing file
    This illegal FileNode will be ignored
ERROR(Predicate4_13): SharedFile sh_file12.s with FileNode(14) from SharedPool(1) includes FileNode(16) that has no existing file
    This illegal FileNode will be ignored

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    1 |—> 2 |—> 1
    DETAILS:
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
        NodeNumber(2)    FileName(file11.inm)Framework(1)   Component(1,1)
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    1 |—> 2 |—> 3 |—> 1
    DETAILS:
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
        NodeNumber(2)    FileName(file11.inm)Framework(1)   Component(1,1)
        NodeNumber(3)    FileName(file11.inl)  Framework(1)  Component(1,1)
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    1 |—> 3 |—> 1
    DETAILS:
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
        NodeNumber(3)    FileName(file11.inl)  Framework(1)  Component(1,1)
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    2 |—> 1 |—> 2
    DETAILS:
        NodeNumber(2)    FileName(file11.inm)Framework(1)   Component(1,1)
        NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
        NodeNumber(2)    FileName(file11.inm)Framework(1)   Component(1,1)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  2 |--> 3 |--> 1 |--> 2
  DETAILS:
      NodeNumber(2)    FileName(file11.inm) Framework(1)  Component(1,1)
      NodeNumber(3)    FileName(file11.inl) Framework(1)  Component(1,1)
      NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
      NodeNumber(2)    FileName(file11.inm) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  3 |--> 1 |--> 2 |--> 3
  DETAILS:
      NodeNumber(3)    FileName(file11.inl) Framework(1)  Component(1,1)
      NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
      NodeNumber(2)    FileName(file11.inm) Framework(1)  Component(1,1)
      NodeNumber(3)    FileName(file11.inl) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  3 |--> 1 |--> 3
  DETAILS:
      NodeNumber(3)    FileName(file11.inl) Framework(1)  Component(1,1)
      NodeNumber(1)    FileName(file11.top) Framework(1)  Component(1,1)
      NodeNumber(3)    FileName(file11.inl) Framework(1)  Component(1,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  4 |--> 7 |--> 8 |--> 11 |--> 12 |--> 4
  DETAILS:
      NodeNumber(4)    FileName(file21.top) Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(8)    FileName(file31.inm)     Framework(3)  Component(3,1)
      NodeNumber(11)   FileName(file12.inm)     Framework(3)  Component(3,2)
      NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
      NodeNumber(4)    FileName(file21.top) Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  4 |--> 7 |--> 11 |--> 12 |--> 4
  DETAILS:
      NodeNumber(4)    FileName(file21.top) Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(11)   FileName(file12.inm)     Framework(3)  Component(3,2)
      NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
      NodeNumber(4)    FileName(file21.top) Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  5 |--> 7 |--> 8 |--> 9 |--> 10 |--> 5
  DETAILS:
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(8)    FileName(file31.inm)     Framework(3)  Component(3,1)
      NodeNumber(9)    FileName(file31.inl) Framework(3)  Component(3,1)
      NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  5 |--> 7 |--> 8 |--> 10 |--> 5
  DETAILS:
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(8)    FileName(file31.inm)     Framework(3)  Component(3,1)
      NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  5 |--> 7 |--> 8 |--> 11 |--> 10 |--> 5
  DETAILS:
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(8)    FileName(file31.inm)     Framework(3)  Component(3,1)
      NodeNumber(11)   FileName(file12.inm)     Framework(3)  Component(3,2)
      NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
  5 |--> 7 |--> 11 |--> 8 |--> 9 |--> 10 |--> 5
  DETAILS:
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)
      NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
      NodeNumber(11)   FileName(file12.inm)     Framework(3)  Component(3,2)
      NodeNumber(8)    FileName(file31.inm)     Framework(3)  Component(3,1)
      NodeNumber(9)    FileName(file31.inl) Framework(3)  Component(3,1)
      NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
      NodeNumber(5)    FileName(file21.inm)     Framework(2)  Component(2,1)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 7 |—> 11 |—> 8 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)    FileName(file12.inm)     Framework(3)  Component(3,2)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 7 |—> 11 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)    FileName(file12.inm)     Framework(3)  Component(3,2)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 7 |—> 11 |—> 12 |—> 8 |—> 9 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)    FileName(file12.inm)     Framework(3)  Component(3,2)
        NodeNumber(12)    FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 7 |—> 11 |—> 12 |—> 8 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)    FileName(file12.inm)     Framework(3)  Component(3,2)
        NodeNumber(12)    FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 8 |—> 9 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 8 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 8 |—> 11 |—> 10 |—> 5
    DETAILS:
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(11)    FileName(file12.inm)     Framework(3)  Component(3,2)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    7 |—> 8 |—> 9 |—> 10 |—> 5 |—> 7
    DETAILS:
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(8)     FileName(file31.inm)     Framework(3)  Component(3,1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)     FileName(file21.inm)     Framework(2)  Component(2,1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    7 |—> 8 |—> 10 |—> 5 |—> 7
    DETAILS:
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3,1)
```

NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 8 |—> 11 |—> 10 |—> 5 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 8 |—> 11 |—> 12 |—> 4 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 8 |—> 11 |—> 12 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 11 |—> 8 |—> 9 |—> 10 |—> 5 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 11 |—> 8 |—> 10 |—> 5 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 11 |—> 10 |—> 5 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 11 |—> 12 |—> 4 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
7 |—> 11 |—> 12 |—> 7
DETAILS:
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    7 |—> 11 |—> 12 |—> 8 |—> 9 |—> 10 |—> 5 |—> 7
    DETAILS:
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(11)    FileName(file12.inm)      Framework(3)  Component(3.2)
        NodeNumber(12)    FileName(file12.inl) Framework(3)  Component(3.2)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    7 |—> 11 |—> 12 |—> 8 |—> 10 |—> 5 |—> 7
    DETAILS:
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(11)    FileName(file12.inm)      Framework(3)  Component(3.2)
        NodeNumber(12)    FileName(file12.inl) Framework(3)  Component(3.2)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 9 |—> 10 |—> 5 |—> 7 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 9 |—> 10 |—> 5 |—> 7 |—> 11 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(11)    FileName(file12.inm)      Framework(3)  Component(3.2)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 9 |—> 10 |—> 5 |—> 7 |—> 11 |—> 12 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(11)    FileName(file12.inm)      Framework(3)  Component(3.2)
        NodeNumber(12)    FileName(file12.inl) Framework(3)  Component(3.2)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 9 |—> 10 |—> 5 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 9 |—> 10 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(9)     FileName(file31.inl) Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 10 |—> 5 |—> 7 |—> 8
    DETAILS:
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
        NodeNumber(10)    FileName(file12.top) Framework(3)  Component(3.2)
        NodeNumber(5)     FileName(file21.inm)      Framework(2)  Component(2.1)
        NodeNumber(7)     FileName(file31.top) Framework(3)  Component(3.1)
        NodeNumber(8)     FileName(file31.inm)      Framework(3)  Component(3.1)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 10 |—> 5 |—> 7 |—> 11 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 10 |—> 5 |—> 7 |—> 11 |—> 12 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 10 |—> 5 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 10 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 10 |—> 5 |—> 7 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 10 |—> 5 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 10 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 12 |—> 4 |—> 7 |—> 8
    DETAILS:
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(4)    FileName(file21.top) Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    8 |—> 11 |—> 12 |—> 7 |—> 8
```

DETAILS:
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
8 |—> 11 |—> 12 |—> 8
DETAILS:
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 5 |—> 7 |—> 8 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 5 |—> 7 |—> 11 |—> 8 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 5 |—> 7 |—> 11 |—> 12 |—> 8 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 5 |—> 8 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 8 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
9 |—> 10 |—> 9
DETAILS:
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 5 |—> 7 |—> 8 |—> 9 |—> 10
DETAILS:
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 8 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 8 |—> 11 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 11 |—> 8 |—> 9 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(9)    FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 11 |—> 8 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 11 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 11 |—> 12 |—> 8 |—> 9 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(9)    FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 7 |—> 11 |—> 12 |—> 8 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(7)    FileName(file31.top) Framework(3)  Component(3,1)
        NodeNumber(11)   FileName(file12.inm)    Framework(3)  Component(3,2)
        NodeNumber(12)   FileName(file12.inl) Framework(3)  Component(3,2)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 5 |—> 8 |—> 9 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
        NodeNumber(5)    FileName(file21.inm)    Framework(2)  Component(2,1)
        NodeNumber(8)    FileName(file31.inm)    Framework(3)  Component(3,1)
        NodeNumber(9)    FileName(file31.inl) Framework(3)  Component(3,1)
        NodeNumber(10)   FileName(file12.top) Framework(3)  Component(3,2)
```

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 5 |—> 8 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(5) | FileName(file21.inm) | Framework(2) | Component(2,1) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 5 |—> 8 |—> 11 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(5) | FileName(file21.inm) | Framework(2) | Component(2,1) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 8 |—> 9 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(9) | FileName(file31.inl) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 8 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 8 |—> 11 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
10 |—> 9 |—> 10
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(9) | FileName(file31.inl) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 |—> 8 |—> 9 |—> 10 |—> 5 |—> 7 |—> 11
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(9) | FileName(file31.inl) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(5) | FileName(file21.inm) | Framework(2) | Component(2,1) |
| NodeNumber(7) | FileName(file31.top) | Framework(3) | Component(3,1) |
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 |—> 8 |—> 10 |—> 5 |—> 7 |—> 11
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(5) | FileName(file21.inm) | Framework(2) | Component(2,1) |
| NodeNumber(7) | FileName(file31.top) | Framework(3) | Component(3,1) |
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 |—> 8 |—> 11
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(8) | FileName(file31.inm) | Framework(3) | Component(3,1) |
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 |—> 10 |—> 5 |—> 7 |—> 8 |—> 11
DETAILS:

| | | | |
|---|---|---|---|
| NodeNumber(11) | FileName(file12.inm) | Framework(3) | Component(3,2) |
| NodeNumber(10) | FileName(file12.top) | Framework(3) | Component(3,2) |
| NodeNumber(5) | FileName(file21.inm) | Framework(2) | Component(2,1) |
| NodeNumber(7) | FileName(file31.top) | Framework(3) | Component(3,1) |

NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 10 ⊢> 5 ⊢> 7 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.iam) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 10 ⊢> 5 ⊢> 8 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.iam) Framework(2) Component(2,1)
NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 10 ⊢> 8 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 4 ⊢> 7 ⊢> 8 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 4 ⊢> 7 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 7 ⊢> 8 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 7 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 8 ⊢> 9 ⊢> 10 ⊢> 5 ⊢> 7 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.iam) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.iam) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.iam) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 8 ⊢> 10 ⊢> 5 ⊢> 7 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
11 ⊢> 12 ⊢> 8 ⊢> 11
DETAILS:
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 4 ⊢> 7 ⊢> 8 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 4 ⊢> 7 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(4) FileName(file21.top) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 7 ⊢> 8 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 7 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 8 ⊢> 9 ⊢> 10 ⊢> 5 ⊢> 7 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(9) FileName(file31.inl) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 8 ⊢> 10 ⊢> 5 ⊢> 7 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(10) FileName(file12.top) Framework(3) Component(3,2)
NodeNumber(5) FileName(file21.inm) Framework(2) Component(2,1)
NodeNumber(7) FileName(file31.top) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
12 ⊢> 8 ⊢> 11 ⊢> 12
DETAILS:
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)
NodeNumber(8) FileName(file31.inm) Framework(3) Component(3,1)
NodeNumber(11) FileName(file12.inm) Framework(3) Component(3,2)
NodeNumber(12) FileName(file12.inl) Framework(3) Component(3,2)

Checking for Predicate 4_12 is complete — STATUS: false

==================================================

Checking for Predicate 4_13: GoodUseOf_top_Files = predicates 4_18, 4_19, and 4_20

Checking for predicate 4_17: NoFileMayIncludeA_top_File

VIOLATION(Predicate4_18): Top File file11.top from Framework(1) is included by file11.inm from Framwork(1)

VIOLATION(Predicate4_18): Top File file11.top from Framework(1) is included by file11.inl from Framwork(1)

VIOLATION(Predicate4_18): Top File file21.top from Framework(2) is included by file12.inl from Framwork(3)

VIOLATION(Predicate4_18): Top File file31.top from Framework(3) is included by file21.top from Framwork(2)

VIOLATION(Predicate4_18): Top File file31.top from Framework(3) is included by file21.inm from Framwork(2)

VIOLATION(Predicate4_18): Top File file31.top from Framework(3) is included by file12.inl from Framwork(3)

VIOLATION(Predicate4_18): Top File file12.top from Framework(3) is included by file31.inm from Framwork(3)

VIOLATION(Predicate4_18): Top File file12.top from Framework(3) is included by file31.inl from Framwork(3)

VIOLATION(Predicate4_18): Top File file12.top from Framework(3) is included by file12.inm from Framwork(3)

Checking for predicate 4_17 is complete —— STATUS: false

==================================================

Checking for predicate 4_18: FromSame_Fr_A_top_FileMayIncludeOnlyIts_intermediate_File

VIOLATION(Predicate4_18): Top File file11.top from Framework(1) includes its Internal File

VIOLATION(Predicate4_18): Top File file21.top from Framework(2) includes File file31.top from Framework(3)

VIOLATION(Predicate4_18): Top File file21.top from Framework(2) includes File file11.inm from Framework(1)

VIOLATION(Predicate4_18): Top File file12.top from Framework(3) includes File file21.inm from Framework(2)

Checking for predicate 4_18 is complete —— STATUS: false

==================================================

Checking for predicate 4_19: A_top_FileMayInclude_shared_FilesFromPermittedSharedPools

VIOLATION(Predicate4_19): Top File file31.top from Framework(3) includes a shared file from a non–accessible SharedPool(1)

Checking for predicate 4_19 is complete —— STATUS: false

==================================================

Checking for Predicate 4_13 is complete — STATUS: false

==================================================

Checking for Predicate 4_14: GoodUseOf_intermediate_Files = predicates 4_21 and 4_22

Checking for predicate 4_20: An_intermediate_fileMayIncludeOnlyIts_internal_FileAndMayIncludeOther_intermediate_FilesFromSame_Fr

VIOLATION(Predicate4_20): Intermediate File file11.inm from Framework(1) includes its top file file11.top

VIOLATION(Predicate4_20): Intermediate File file21.inm from Framework(2 includes file31.inm from Framework(3)

VIOLATION(Predicate4_20): Intermediate File file21.inm from Framework(2 includes file31.top from Framework(3)

ERROR(Predicate4_20): Intermediate File file21.inm includes FileNode(30) that is not found in any Framework or SharedPool

VIOLATION(Predicate4_20): Intermediate File file12.inm from Framework(3) includes its top file file12.top

Checking for predicate 4_20 is complete —— STATUS: false

==================================================

Checking for Predicate 4_14 is complete —— STATUS: false

==================================================

Checking for Predicate 4_15: GoodUseOf_internal_Files = predicates 4_23 and 4_24

Checking for predicate 4_21: An_internal_FileMayNotIncludeAnyFile

VIOLATION(Predicate 4_21): InternalFile file11.inl Includes FileNode 1

VIOLATION(Predicate 4_21): InternalFile file21.inl Includes FileNode 8

VIOLATION(Predicate 4_21): InternalFile file31.inl Includes FileNode 10

VIOLATION(Predicate 4_21): InternalFile file12.inl Includes FileNode 7

VIOLATION(Predicate 4_21): InternalFile file12.inl Includes FileNode 4

VIOLATION(Predicate 4_21): InternalFile file12.inl Includes FileNode 8

Checking for predicate 4_21 is complete — STATUS:false

Checking for predicate 4_22: NoFileMayIncludeAn_Internal_FileExceptIts_intermediate_File

VIOLATION(Predicate 4_22): Internal File file11.inl is included by file11.top from Framework(1) — Component(1,1)

VIOLATION(Predicate 4_22): Internal File file31.inl is included by file12.top from Framework(3) — Component(3,2)

Checking for predicate 4_22 is complete — STATUS:false

Checking for Predicate 4_15 is complete — STATUS: false

==========================================================

Checking for Predicate 4_11 is complete — STATUS: false

==========================================================

Checking for Predicate 4_12: InclusionLoopsMayNotExist

Checking for Predicate 4_12 is complete — STATUS: false

==========================================================

Checking for Predicate 4_8 is complete — STATUS: false

==========================================================

Checking for GoodFileStructure_predicate4_5 is complete — STATUS:false

==========================================================

# Appendix A4

# A Prototype For CO–Toolkits' File Structures: Example#3 – ACE Sample File Structure

## A4.1 Overview

This Appendix presents the third example that has been used to demonstrate our prototype for CO–Toolkits' file structures. This example has been based on a sample file structure from ACE. Section A4.3 presents the rules violations reported by this prototype for this example. For convenience, we have replaced the .cpp, .h, and .i filename extensions used in ACE by .top, .inm, and .inl.

## A4.2 Example#3: ACE Sample File Structure

```
FILE STRUCTURE: As Extracted From The Current Example
Framework(1)
    Component(1,1)
        FileNode(1):  IO_Cntl_Msg.top   Includes FileNodes:
        FileNode(2):  IO_Cntl_Msg.inm   Includes FileNodes:
        FileNode(3):  IO_Cntl_Msg.inl   Includes FileNodes:
    Component(1,2)
        FileNode(4):  Map_Manager.top   Includes FileNodes:   56  110 155 5   6
        FileNode(5):  Map_Manager.inm   Includes FileNodes:   116 6   4
        FileNode(6):  Map_Manager.inl   Includes FileNodes:
    Component(1,3)
        FileNode(7):  Message_Block.top Includes FileNodes:   8   9
        FileNode(8):  Message_Block.inm     Includes FileNodes:   116 110 9
        FileNode(9):  Message_Block.inl Includes FileNodes:
    Component(1,4)
        FileNode(10): Message_Queue.top Includes FileNodes:   11  12
        FileNode(11): Message_Queue.inm     Includes FileNodes:   8   2   83  12  10
        FileNode(12): Message_Queue.inl Includes FileNodes:
    Component(1,5)
        FileNode(13): Module.top    Includes FileNodes:   14  23  15
        FileNode(14): Module.inm    Includes FileNodes:   116 26  15  13
        FileNode(15): Module.inl    Includes FileNodes:
    Component(1,6)
        FileNode(16): Multiplexor.top   Includes FileNodes:   17  18
```

FileNode(17): Multiplexor.inm Includes FileNodes: 14 5 18
FileNode(18): Multiplexor.inl Includes FileNodes:
Component(1,7)
FileNode(19): Stream.top Includes FileNodes: 14 20 23 21
FileNode(20): Stream.inm Includes FileNodes: 17 116 2 8 143 14 21 19
FileNode(21): Stream.inl Includes FileNodes:
Component(1,8)
FileNode(22): Stream_Modules.top Includes FileNodes: 17 18
FileNode(23): Stream_Modules.inm Includes FileNodes: 26 24 22
FileNode(24): Stream_Modules.inl Includes FileNodes:
Component(1,9)
FileNode(25): Task.top Includes FileNodes: 26 14 155 27
FileNode(26): Task.inm Includes FileNodes: 159 68 27
FileNode(27): Task.inl Includes FileNodes:
Framework(2)
Component(2,1)
FileNode(28): CORBA_Handler.top Includes FileNodes: 29 30
FileNode(29): CORBA_Handler.inm Includes FileNodes: 155 157 158 30
FileNode(30): CORBA_Handler.inl Includes FileNodes:
Component(2,2)
FileNode(31): CORBA_Ref.top Includes FileNodes: 32 33
FileNode(32): CORBA_Ref.inm Includes FileNodes: 33 31
FileNode(33): CORBA_Ref.inl Includes FileNodes:
Framework(3)
Component(3,1)
FileNode(34): Set.top Includes FileNodes: 35 36
FileNode(35): Set.inm Includes FileNodes: 116 36 34
FileNode(36): Set.inl Includes FileNodes:
Component(3,2)
FileNode(37): Stack.top Includes FileNodes: 38 39
FileNode(38): Stack.inm Includes FileNodes: 116 39 37
FileNode(39): Stack.inl Includes FileNodes:
Component(3,3)
FileNode(40): SString.top Includes FileNodes: 110 155 41 42
FileNode(41): SString.inm Includes FileNodes: 116 42
FileNode(42): SString.inl Includes FileNodes:
Framework(4)
Component(4,1)
FileNode(43): Activation_Queue.top Includes FileNodes: 44
FileNode(44): Activation_Queue.inm Includes FileNodes: 56 11 50
FileNode(45): Activation_Queue.inl Includes FileNodes:
Component(4,2)
FileNode(46): Future.top Includes FileNodes: 47
FileNode(47): Future.inm Includes FileNodes: 56 46
FileNode(48): Future.inl Includes FileNodes:
Component(4,3)
FileNode(49): Method_Object.top Includes FileNodes: 50
FileNode(50): Method_Object.inm Includes FileNodes: 119
FileNode(51): Method_Object.inl Includes FileNodes:
Component(4,4)
FileNode(52): Process_Manager.top Includes FileNodes: 53 54
FileNode(53): Process_Manager.inm Includes FileNodes: 56 54
FileNode(54): Process_Manager.inl Includes FileNodes:
Component(4,5)
FileNode(55): Synch.top Includes FileNodes: 65 56 143 57
FileNode(56): Synch.inm Includes FileNodes: 116 149 57 62
FileNode(57): Synch.inl Includes FileNodes:
Component(4,6)
FileNode(58): Synch_Options.top Includes FileNodes: 59 116
FileNode(59): Synch_Options.inm Includes FileNodes:
FileNode(60): Synch_Options.inl Includes FileNodes:
Component(4,7)
FileNode(61): Synch_T.top Includes FileNodes: 65 143 62 63
FileNode(62): Synch_T.inm Includes FileNodes: 116 56 63 61
FileNode(63): Synch_T.inl Includes FileNodes: 65
Component(4,8)
FileNode(64): Thread.top Includes FileNodes: 65 66
FileNode(65): Thread.inm Includes FileNodes: 116 66
FileNode(66): Thread.inl Includes FileNodes:
Component(4,9)
FileNode(67): Thread_Manager.top Includes FileNodes: 68 69
FileNode(68): Thread_Manager.inm Includes FileNodes: 65 56 69
FileNode(69): Thread_Manager.inl Includes FileNodes:
Component(4,10)
FileNode(70): Token.top Includes FileNodes: 65 143 71 72

```
        FileNode(71): Token.inm   Includes FileNodes:   56  72
        FileNode(72): Token.inl Includes FileNodes:
Framework(5)
    Component(5,1)
        FileNode(73): Acceptor.top  Includes FileNodes:   116 74
        FileNode(74): Acceptor.inm  Includes FileNodes:   155 159 86  83  75  73
        FileNode(75): Acceptor.inl  Includes FileNodes:
    Component(5,2)
        FileNode(76): Connector.top Includes FileNodes:   77
        FileNode(77): Connector.inm Includes FileNodes:   155 159 146 5   86  83  78  76
        FileNode(78): Connector.inl  Includes FileNodes:
    Component(5,3)
        FileNode(79): Dynamic_Service.top   Includes FileNodes:   155 160 80  11
        FileNode(80): Dynamic_Service.inm   Includes FileNodes:   79
        FileNode(81): Dynamic_Service.inl   Includes FileNodes:
    Component(5,4)
        FileNode(82): Strategies.top  Includes FileNodes:   134 137 83
        FileNode(83): Strategies.inm  Includes FileNodes:   122
        FileNode(84): Strategies.inl  Includes FileNodes:
    Component(5,5)
        FileNode(85): Svc_Handler.top   Includes FileNodes:   86  113 87
        FileNode(86): Svc_Handler.inm   Includes FileNodes:   59  26  155 62  87  85
        FileNode(87): Svc_Handler.inl   Includes FileNodes:
Framework(6)
    Component(6,1)
        FileNode(88): FIFO_Send_Msg.top    Includes FileNodes:
        FileNode(89): FIFO_Send_Msg.inm    Includes FileNodes:
        FileNode(90): FIFO_Send_Msg.inl    Includes FileNodes:
    Component(6,2)
        FileNode(91): SOCK_Acceptor.top    Includes FileNodes:
        FileNode(92): SOCK_Acceptor.inm    Includes FileNodes:
        FileNode(93): SOCK_Acceptor.inl Includes FileNodes:
    Component(6,3)
        FileNode(94): SOCK_Connector.top   Includes FileNodes:
        FileNode(95): SOCK_Connector.inm   Includes FileNodes:
        FileNode(96): SOCK_Connector.inl   Includes FileNodes:
    Component(6,4)
        FileNode(97): SOCK_Stream.top Includes FileNodes:
        FileNode(98): SOCK_Stream.inm Includes FileNodes:
        FileNode(99): SOCK_Stream.inl Includes FileNodes:
    Component(6,5)
        FileNode(100):     SPIPE_Connector.top    Includes FileNodes:
        FileNode(101):     SPIPE_Connector.inm    Includes FileNodes:
        FileNode(102):     SPIPE_Connector.inl    Includes FileNodes:
Framework(7)
    Component(7,1)
        FileNode(103):     Log_Msg.top  Includes FileNodes:   154 116 117 65  56  101 89
        FileNode(104):     Log_Msg.inm  Includes FileNodes:   107 105
        FileNode(105):     Log_Msg.inl  Includes FileNodes:
    Component(7,2)
        FileNode(106):     Log_Record.top   Includes FileNodes:   107
        FileNode(107):     Log_Record.inm   Includes FileNodes:   116 108
        FileNode(108):     Log_Record.inl   Includes FileNodes:
Framework(8)
    Component(8,1)
        FileNode(109):     Malloc.top    Includes FileNodes:
        FileNode(110):     Malloc.inm    Includes FileNodes:
        FileNode(111):     Malloc.inl    Includes FileNodes:
Framework(9)
    Component(9,1)
        FileNode(112):     Dynamic.top   Includes FileNodes:
        FileNode(113):     Dynamic.inm   Includes FileNodes:
        FileNode(114):     Dynamic.inl   Includes FileNodes:
Framework(10)
    Component(10,1)
        FileNode(115):     ACE.top  Includes FileNodes:
        FileNode(116):     ACE.inm  Includes FileNodes:
        FileNode(117):     ACE.inl  Includes FileNodes:
    Component(10,2)
        FileNode(118):     OS.top    Includes FileNodes:
        FileNode(119):     OS.inm    Includes FileNodes:
        FileNode(120):     OS.inl    Includes FileNodes:
Framework(11)
    Component(11,1)
        FileNode(121):     Event_Handler.top  Includes FileNodes:   122 8   123
```

```
            FileNode(122):   Event_Handler.inm Includes FileNodes:    116 123
            FileNode(123):   Event_Handler.inl  Includes FileNodes:   122
        Component(11,2)
            FileNode(124):   Event_Handler_T.top    Includes FileNodes:   125 126
            FileNode(125):   Event_Handler_T.inm    Includes FileNodes:   122 126
            FileNode(126):   Event_Handler_T.inl    Includes FileNodes:
        Component(11,3)
            FileNode(127):   Handle_Set.top    Includes FileNodes:   128 129
            FileNode(128):   Handle_Set.inm    Includes FileNodes:   116 129
            FileNode(129):   Handle_Set.inl Includes FileNodes:
        Component(11,4)
            FileNode(130):   Proactor.top   Includes FileNodes:   131 132
            FileNode(131):   Proactor.inm   Includes FileNodes:   119 146 122 132
            FileNode(132):   Proactor.inl   Includes FileNodes:
        Component(11,5)
            FileNode(133):   Reactor.top    Includes FileNodes:   62  92  95  134
            FileNode(134):   Reactor.inm    Includes FileNodes:   128 146 140 65  71  156 98  152 135
            FileNode(135):   Reactor.inl    Includes FileNodes:
        Component(11,6)
            FileNode(136):   ReactorEx.top Includes FileNodes:   137 138
            FileNode(137):   ReactorEx.inm Includes FileNodes:   143 146 122 11  71  56  152 138
            FileNode(138):   ReactorEx.inl  Includes FileNodes:
        Component(11,7)
            FileNode(139):   Signal.top    Includes FileNodes:   140 141
            FileNode(140):   Signal.inm    Includes FileNodes:   56  122 35  141
            FileNode(141):   Signal.inl Includes FileNodes:
        Component(11,8)
            FileNode(142):   Time_Value.top    Includes FileNodes:   116 143 144
            FileNode(143):   Time_Value.inm    Includes FileNodes:   117 144 56  147
            FileNode(144):   Time_Value.inl    Includes FileNodes:
        Component(11,9)
            FileNode(145):   Timer_Queue.top   Includes FileNodes:   146 147
            FileNode(146):   Timer_Queue.inm   Includes FileNodes:
            FileNode(147):   Timer_Queue.inl   Includes FileNodes:
    Framework(12)
        Component(12,1)
            FileNode(148):   SV_Semaphore_Complex.top Includes FileNodes:
            FileNode(149):   SV_Semaphore_Complex.inm     Includes FileNodes:
            FileNode(150):   SV_Semaphore_Complex.inl  Includes FileNodes:
    Framework(13)
        Component(13,1)
            FileNode(151):   Local_Tokens.top   Includes FileNodes:
            FileNode(152):   Local_Tokens.inm   Includes FileNodes:
            FileNode(153):   Local_Tokens.inl   Includes FileNodes:
    SharedPool(1)
            FileNode(154):   config.s   Includes FileNodes:
            FileNode(155):   Service_Config.s   Includes FileNodes:
            FileNode(156):   Pipe.s     Includes FileNodes:
            FileNode(157):   CORBA.sIncludes FileNodes:
            FileNode(158):   daemon.s Includes FileNodes:
            FileNode(159):   Service_Object.s   Includes FileNodes:
            FileNode(160):   Service_Repository.s    Includes FileNodes:
        Permitted Frameworks to this SharedPool are:    Framework(1) Framework(2) Framework(10)
    FILE STRUCTURE display is DONE
    ==========================================================
```

# A4.3 Example#3: ACE Sample Output Report

```
Checking for GoodFileStructure_predicate4_5: GoodFileStructure = predicates 4_6, 4_7, and 4_8
Checking for Predicate 4_6: OnlyFourFileCategories
Checking for Predicate 4_6 is complete --- STATUS: true
==========================================================
Checking for Predicate 4_7: GoodDistinctionOfFiles= Predicates 4_9, and 4_10
Checking for Predicate 4_9: Fr_SetsOfFilesShouldBeDistinct
Checking for Predicate 4_9 is complete --- STATUS: true
==========================================================
Checking for Predicate 4_10: ComponentsMainFilesShouldBeDistinct
Checking for Predicate 4_10 is complete --- STATUS: true
==========================================================
Checking for Predicate 4_7 is complete --- STATUS: true
```

```
=======================================================================
Checking for Predicate 4_8: GoodFileInclusionRelation = predicates 4_11, 4_12, and 4_13
Checking for Predicate 4_11: GoodUseOfFiles = predicates 4_14, 4_15, 4_16, and 4_17
Checking for Predicate 4_12: InclusionLoopsMayNotExist
FILE STRUCTURE: As Extracted From The Current Example
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    4 |--> 5 |--> 4
    DETAILS:
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
        NodeNumber(5)   FileName(Map_Manager.inm)    Framework(1) Component(1,2)
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |--> 4 |--> 5
    DETAILS:
        NodeNumber(5)   FileName(Map_Manager.inm)    Framework(1) Component(1,2)
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
        NodeNumber(5)   FileName(Map_Manager.inm)    Framework(1) Component(1,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |--> 11 |--> 10
    DETAILS:
        NodeNumber(10)  FileName(Message_Queue.top)  Framework(1) Component(1,4)
        NodeNumber(11)  FileName(Message_Queue.inm)  Framework(1) Component(1,4)
        NodeNumber(10)  FileName(Message_Queue.top)  Framework(1) Component(1,4)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    11 |--> 10 |--> 11
    DETAILS:
        NodeNumber(11)  FileName(Message_Queue.inm)  Framework(1) Component(1,4)
        NodeNumber(10)  FileName(Message_Queue.top)  Framework(1) Component(1,4)
        NodeNumber(11)  FileName(Message_Queue.inm)  Framework(1) Component(1,4)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    13 |--> 14 |--> 13
    DETAILS:
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    13 |--> 23 |--> 22 |--> 17 |--> 14 |--> 13
    DETAILS:
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    14 |--> 13 |--> 14
    DETAILS:
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    14 |--> 13 |--> 23 |--> 22 |--> 17 |--> 14
    DETAILS:
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    17 |--> 14 |--> 13 |--> 23 |--> 22 |--> 17
    DETAILS:
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    19 |--> 20 |--> 19
    DETAILS:
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    20 |—> 19 |—> 20
    DETAILS:
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    22 |—> 17 |—> 14 |—> 13 |—> 23 |—> 22
    DETAILS:
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    23 |—> 22 |—> 17 |—> 14 |—> 13 |—> 23
    DETAILS:
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    31 |—> 32 |—> 31
    DETAILS:
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    32 |—> 31 |—> 32
    DETAILS:
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    34 |—> 35 |—> 34
    DETAILS:
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    35 |—> 34 |—> 35
    DETAILS:
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    37 |—> 38 |—> 37
    DETAILS:
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    38 |—> 37 |—> 38
    DETAILS:
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    46 |—> 47 |—> 46
    DETAILS:
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    47 |—> 46 |—> 47
    DETAILS:
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    56 |—> 62 |—> 56
    DETAILS:
```

```
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    56 |—> 62 |—> 61 |—> 143 |—> 56
    DETAILS:
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)   Framework(11)    Component(11,8)
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    61 |—> 62 |—> 61
    DETAILS:
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    61 |—> 143 |—> 56 |—> 62 |—> 61
    DETAILS:
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)   Framework(11)    Component(11,8)
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 56 |—> 62
    DETAILS:
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 61 |—> 62
    DETAILS:
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 61 |—> 143 |—> 56 |—> 62
    DETAILS:
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)   Framework(11)    Component(11,8)
        NodeNumber(56)   FileName(Synch.inm)    Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    73 |—> 74 |—> 73
    DETAILS:
        NodeNumber(73)   FileName(Acceptor.top) Framework(5) Component(5,1)
        NodeNumber(74)   FileName(Acceptor.inm) Framework(5) Component(5,1)
        NodeNumber(73)   FileName(Acceptor.top) Framework(5) Component(5,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    74 |—> 73 |—> 74
    DETAILS:
        NodeNumber(74)   FileName(Acceptor.inm) Framework(5) Component(5,1)
        NodeNumber(73)   FileName(Acceptor.top) Framework(5) Component(5,1)
        NodeNumber(74)   FileName(Acceptor.inm) Framework(5) Component(5,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    76 |—> 77 |—> 76
    DETAILS:
        NodeNumber(76)   FileName(Connector.top)     Framework(5) Component(5,2)
        NodeNumber(77)   FileName(Connector.inm)     Framework(5) Component(5,2)
        NodeNumber(76)   FileName(Connector.top)     Framework(5) Component(5,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    77 |—> 76 |—> 77
    DETAILS:
        NodeNumber(77)   FileName(Connector.inm)     Framework(5) Component(5,2)
        NodeNumber(76)   FileName(Connector.top)     Framework(5) Component(5,2)
        NodeNumber(77)   FileName(Connector.inm)     Framework(5) Component(5,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    79 |—> 80 |—> 79
    DETAILS:
        NodeNumber(79)   FileName(Dynamic_Service.top)   Framework(5) Component(5,3)
        NodeNumber(80)   FileName(Dynamic_Service.inm)   Framework(5) Component(5,3)
        NodeNumber(79)   FileName(Dynamic_Service.top)   Framework(5) Component(5,3)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    80 |—> 79 |—> 80
    DETAILS:
        NodeNumber(80)   FileName(Dynamic_Service.inm)  Framework(5) Component(5,3)
        NodeNumber(79)   FileName(Dynamic_Service.top)  Framework(5) Component(5,3)
        NodeNumber(80)   FileName(Dynamic_Service.inm)  Framework(5) Component(5,3)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    85 |—> 86 |—> 85
    DETAILS:
        NodeNumber(85)   FileName(Svc_Handler.top)  Framework(5) Component(5,5)
        NodeNumber(86)   FileName(Svc_Handler.inm)  Framework(5) Component(5,5)
        NodeNumber(85)   FileName(Svc_Handler.top)  Framework(5) Component(5,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    86 |—> 85 |—> 86
    DETAILS:
        NodeNumber(86)   FileName(Svc_Handler.inm)  Framework(5) Component(5,5)
        NodeNumber(85)   FileName(Svc_Handler.top)  Framework(5) Component(5,5)
        NodeNumber(86)   FileName(Svc_Handler.inm)  Framework(5) Component(5,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    122 |—> 123 |—> 122
    DETAILS:
        NodeNumber(122)  FileName(Event_Handler.inm)    Framework(11)    Component(11,1)
        NodeNumber(123)  FileName(Event_Handler.inl) Framework(11)    Component(11,1)
        NodeNumber(122)  FileName(Event_Handler.inm)    Framework(11)    Component(11,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    123 |—> 122 |—> 123
    DETAILS:
        NodeNumber(123)  FileName(Event_Handler.inl) Framework(11)    Component(11,1)
        NodeNumber(122)  FileName(Event_Handler.inm)    Framework(11)    Component(11,1)
        NodeNumber(123)  FileName(Event_Handler.inl) Framework(11)    Component(11,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    143 |—> 56 |—> 62 |—> 61 |—> 143
    DETAILS:
        NodeNumber(143)  FileName(Time_Value.inm)  Framework(11)    Component(11,8)
        NodeNumber(56)   FileName(Synch.inm)   Framework(4) Component(4,5)
        NodeNumber(62)   FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)   FileName(Synch_T.top) Framework(4) Component(4,7)
        NodeNumber(143)  FileName(Time_Value.inm)  Framework(11)    Component(11,8)
Checking for Predicate 4_12 is complete —— STATUS: false
===========================================================================
Checking for Predicate 4_13: GoodUseOf_top_Files = predicates 4_18, 4_19, and 4_20
Checking for predicate 4_17: NoFileMayIncludeA_top_File
VIOLATION(Predicate4_18): Top File Map_Manager.top from Framework(1) is included by Map_Manager.inm from Framwork(1)
VIOLATION(Predicate4_18): Top File Message_Queue.top from Framework(1) is included by Message_Queue.inm from Framwork(1)
VIOLATION(Predicate4_18): Top File Module.top from Framework(1) is included by Module.inm from Framwork(1)
VIOLATION(Predicate4_18): Top File Stream.top from Framework(1) is included by Stream.inm from Framwork(1)
VIOLATION(Predicate4_18): Top File Stream_Modules.top from Framework(1) is included by Stream_Modules.inm from Framwork(1)
VIOLATION(Predicate4_18): Top File CORBA_Ref.top from Framework(2) is included by CORBA_Ref.inm from Framwork(2)
VIOLATION(Predicate4_18): Top File Set.top from Framework(3) is included by Set.inm from Framwork(3)
VIOLATION(Predicate4_18): Top File Stack.top from Framework(3) is included by Stack.inm from Framwork(3)
VIOLATION(Predicate4_18): Top File Future.top from Framework(4) is included by Future.inm from Framwork(4)
VIOLATION(Predicate4_18): Top File Synch_T.top from Framework(4) is included by Synch_T.inm from Framwork(4)
VIOLATION(Predicate4_18): Top File Acceptor.top from Framework(5) is included by Acceptor.inm from Framwork(5)
VIOLATION(Predicate4_18): Top File Connector.top from Framework(5) is included by Connector.inm from Framwork(5)
VIOLATION(Predicate4_18): Top File Dynamic_Service.top from Framework(5) is included by Dynamic_Service.inm from Framwork(5)
VIOLATION(Predicate4_18): Top File Svc_Handler.top from Framework(5) is included by Svc_Handler.inm from Framwork(5)
Checking for predicate 4_17 is complete —— STATUS: false
===========================================================================
Checking for predicate 4_18: FromSame_Fr_A_top_FileMayIncludeOnlyIts_intermediate_File
VIOLATION(Predicate4_18): Top File Map_Manager.top from Framework(1) includes File Synch.inm from Framework(4)
VIOLATION(Predicate4_18): Top File Map_Manager.top from Framework(1) includes File Malloc.inm from Framework(8)
VIOLATION(Predicate4_18): Top File Map_Manager.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Message_Block.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Message_Queue.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Module.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Multiplexor.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Stream.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File Task.top from Framework(1) includes its Internal File
VIOLATION(Predicate4_18): Top File CORBA_Handler.top from Framework(2) includes its Internal File
VIOLATION(Predicate4_18): Top File CORBA_Ref.top from Framework(2) includes its Internal File
VIOLATION(Predicate4_18): Top File Set.top from Framework(3) includes its Internal File
VIOLATION(Predicate4_18): Top File Stack.top from Framework(3) includes its Internal File
VIOLATION(Predicate4_18): Top File SString.top from Framework(3) includes File Malloc.inm from Framework(8)
```

VIOLATION(Predicate4_18): Top File SString.top from Framework(3) includes its Internal File
VIOLATION(Predicate4_18): Top File Process_Manager.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Synch.top from Framework(4) includes File Time_Value.inm from Framework(11)
VIOLATION(Predicate4_18): Top File Synch.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Synch_Options.top from Framework(4) includes File ACE.inm from Framework(10)
VIOLATION(Predicate4_18): Top File Synch_T.top from Framework(4) includes File Time_Value.inm from Framework(11)
VIOLATION(Predicate4_18): Top File Synch_T.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Thread.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Thread_Manager.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Token.top from Framework(4) includes File Time_Value.inm from Framework(11)
VIOLATION(Predicate4_18): Top File Token.top from Framework(4) includes its Internal File
VIOLATION(Predicate4_18): Top File Acceptor.top from Framework(5) includes File ACE.inm from Framework(10)
VIOLATION(Predicate4_18): Top File Dynamic_Service.top from Framework(5) includes File Message_Queue.inm from Framework(1)
VIOLATION(Predicate4_18): Top File Strategies.top from Framework(5) includes File Reactor.inm from Framework(11)
VIOLATION(Predicate4_18): Top File Strategies.top from Framework(5) includes File ReactorEx.inm from Framework(11)
VIOLATION(Predicate4_18): Top File Svc_Handler.top from Framework(5) includes File Dynamic.inm from Framework(9)
VIOLATION(Predicate4_18): Top File Svc_Handler.top from Framework(5) includes its Internal File
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File ACE.inm from Framework(10)
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File ACE.inl from Framework(10)
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File Thread.inm from Framework(4)
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File Synch.inm from Framework(4)
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File SPIPE_Connector.inm from Framework(6)
VIOLATION(Predicate4_18): Top File Log_Msg.top from Framework(7) includes File FIFO_Send_Msg.inm from Framework(6)
VIOLATION(Predicate4_18): Top File Event_Handler.top from Framework(11) includes File Message_Block.inm from Framework(1)
VIOLATION(Predicate4_18): Top File Event_Handler.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Event_Handler_T.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Handle_Set.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Proactor.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Reactor.top from Framework(11) includes File Synch_T.inm from Framework(4)
VIOLATION(Predicate4_18): Top File Reactor.top from Framework(11) includes File SOCK_Acceptor.inm from Framework(6)
VIOLATION(Predicate4_18): Top File Reactor.top from Framework(11) includes File SOCK_Connector.inm from Framework(6)
VIOLATION(Predicate4_18): Top File ReactorEx.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Signal.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Time_Value.top from Framework(11) includes File ACE.inm from Framework(10)
VIOLATION(Predicate4_18): Top File Time_Value.top from Framework(11) includes its Internal File
VIOLATION(Predicate4_18): Top File Timer_Queue.top from Framework(11) includes its Internal File
Checking for predicate 4_18 is complete —– STATUS: false

===========================================================

Checking for predicate 4_19: A_top_FileMayInclude_shared_FilesFromPermittedSharedPools
VIOLATION(Predicate4_19): Top File SString.top from Framework(3) includes a shared file from a non-accessible SharedPool(1)
VIOLATION(Predicate4_19): Top File Dynamic_Service.top from Framework(5) includes a shared file from a non-accessible SharedPool(1)
VIOLATION(Predicate4_19): Top File Dynamic_Service.top from Framework(5) includes a shared file from a non-accessible SharedPool(1)
VIOLATION(Predicate4_19): Top File Log_Msg.top from Framework(7) includes a shared file from a non-accessible SharedPool(1)
Checking for predicate 4_19 is complete —– STATUS: false

===========================================================

Checking for Predicate 4_13 is complete — STATUS: false

===========================================================

Checking for Predicate 4_14: GoodUseOf_intermediate_Files = predicates 4_21 and 4_22
Checking for predicate 4_20: An_intermediate_fileMayIncludeOnlyIts_internal_FileAndMayIncludeOther_intermediate_FilesFrom-Same_Fr
VIOLATION(Predicate4_20): Intermediate File Map_Manager.inm from Framework(1 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Map_Manager.inm from Framework(1) includes its top file Map_Manager.top
VIOLATION(Predicate4_20): Intermediate File Message_Block.inm from Framework(1 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Message_Block.inm from Framework(1 includes Malloc.inm from Framework(8)
VIOLATION(Predicate4_20): Intermediate File Message_Queue.inm from Framework(1 includes Strategies.inm from Framework(5)
VIOLATION(Predicate4_20): Intermediate File Message_Queue.inm from Framework(1) includes its top file Message_Queue.top
VIOLATION(Predicate4_20): Intermediate File Module.inm from Framework(1 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Module.inm from Framework(1) includes its top file Module.top
VIOLATION(Predicate4_20): Intermediate File Stream.inm from Framework(1 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Stream.inm from Framework(1 includes Time_Value.inm from Framework(11)
VIOLATION(Predicate4_20): Intermediate File Stream.inm from Framework(1) includes its top file Stream.top
VIOLATION(Predicate4_20): Intermediate File Stream_Modules.inm from Framework(1) includes its top file Stream_Modules.top
VIOLATION(Predicate4_20): Intermediate file Task.inm from Framework(1) includes Service_Object.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File Task.inm from Framework(1 includes Thread_Manager.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate file CORBA_Handler.inm from Framework(2) includes Service_Config.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate file CORBA_Handler.inm from Framework(2) includes CORBA.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate file CORBA_Handler.inm from Framework(2) includes daemon.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File CORBA_Ref.inm from Framework(2) includes its top file CORBA_Ref.top
VIOLATION(Predicate4_20): Intermediate File Set.inm from Framework(3 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Set.inm from Framework(3) includes its top file Set.top
VIOLATION(Predicate4_20): Intermediate File Stack.inm from Framework(3 includes ACE.inm from Framework(10)

VIOLATION(Predicate4_20): Intermediate File Stack.inm from Framework(3) includes its top file Stack.top
VIOLATION(Predicate4_20): Intermediate File SString.inm from Framework(3 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Activation_Queue.inm from Framework(4 includes Message_Queue.inm from Framework(1)
VIOLATION(Predicate4_20): Intermediate File Future.inm from Framework(4) includes its top file Future.top
VIOLATION(Predicate4_20): Intermediate File Method_Object.inm from Framework(4 includes OS.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Synch.inm from Framework(4 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Synch.inm from Framework(4 includes SV_Semaphore_Complex.inm from Framework(12)
VIOLATION(Predicate4_20): Intermediate File Synch_T.inm from Framework(4 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Synch_T.inm from Framework(4) includes its top file Synch_T.top
VIOLATION(Predicate4_20): Intermediate File Thread.inm from Framework(4 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate file Acceptor.inm from Framework(5) includes Service_Config.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate file Acceptor.inm from Framework(5) includes Service_Object.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File Acceptor.inm from Framework(5) includes its top file Acceptor.top
VIOLATION(Predicate4_20): Intermediate file Connector.inm from Framework(5) includes Service_Config.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate file Connector.inm from Framework(5) includes Service_Object.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File Connector.inm from Framework(5 includes Timer_Queue.inm from Framework(11)
VIOLATION(Predicate4_20): Intermediate File Connector.inm from Framework(5 includes Map_Manager.inm from Framework(1)
VIOLATION(Predicate4_20): Intermediate File Connector.inm from Framework(5) includes its top file Connector.top
VIOLATION(Predicate4_20): Intermediate File Dynamic_Service.inm from Framework(5) includes its top file Dynamic_Service.top
VIOLATION(Predicate4_20): Intermediate File Strategies.inm from Framework(5 includes Event_Handler.inm from Framework(11)
VIOLATION(Predicate4_20): Intermediate File Svc_Handler.inm from Framework(5 includes Synch_Options.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File Svc_Handler.inm from Framework(5 includes Task.inm from Framework(1)
VIOLATION(Predicate4_20): Intermediate file Svc_Handler.inm from Framework(5) includes Service_Config.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File Svc_Handler.inm from Framework(5 includes Synch_T.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File Svc_Handler.inm from Framework(5) includes its top file Svc_Handler.top
VIOLATION(Predicate4_20): Intermediate File Log_Record.inm from Framework(7 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Event_Handler.inm from Framework(11 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Handle_Set.inm from Framework(11 includes ACE.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Proactor.inm from Framework(11 includes OS.inm from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Reactor.inm from Framework(11 includes Thread.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File Reactor.inm from Framework(11 includes Token.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate file Reactor.inm from Framework(11) includes Pipe.s from SharedPool(1)
VIOLATION(Predicate4_20): Intermediate File Reactor.inm from Framework(11 includes SOCK_Stream.inm from Framework(6)
VIOLATION(Predicate4_20): Intermediate File Reactor.inm from Framework(11 includes Local_Tokens.inm from Framework(13)
VIOLATION(Predicate4_20): Intermediate File ReactorEx.inm from Framework(11 includes Message_Queue.inm from Framework(1)
VIOLATION(Predicate4_20): Intermediate File ReactorEx.inm from Framework(11 includes Token.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File ReactorEx.inm from Framework(11 includes Synch.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File ReactorEx.inm from Framework(11 includes Local_Tokens.inm from Framework(13)
VIOLATION(Predicate4_20): Intermediate File Signal.inm from Framework(11 includes Synch.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File Signal.inm from Framework(11 includes Set.inm from Framework(3)
VIOLATION(Predicate4_20): Intermediate File Time_Value.inm from Framework(11 includes ACE.inl from Framework(10)
VIOLATION(Predicate4_20): Intermediate File Time_Value.inm from Framework(11 includes Synch.inm from Framework(4)
VIOLATION(Predicate4_20): Intermediate File Time_Value.inm from Framework(11) includes the non-intemediate Timer_Queue.inl from the same framework
Checking for predicate 4_20 is complete —– STATUS: false
==========================================================
Checking for Predicate 4_14 is complete —– STATUS: false
==========================================================
Checking for Predicate 4_15: GoodUseOf_internal_Files = predicates 4_23 and 4_24
Checking for predicate 4_21: An_internal_FileMayNotIncludeAnyFile
VIOLATION(Predicate 4_21): InternalFile Synch_T.inl Includes FileNode 65
VIOLATION(Predicate 4_21): InternalFile Event_Handler.inl Includes FileNode 122
Checking for predicate 4_21 is complete — STATUS:false
Checking for predicate 4_22: NoFileMayIncludeAn_Internal_FileExceptIts_intermediate_File
VIOLATION(Predicate 4_22): Internal File Map_Manager.inl is included by Map_Manager.top from Framework(1) — Component(1,2)
VIOLATION(Predicate 4_22): Internal File Message_Block.inl is included by Message_Block.top from Framework(1) — Component(1,3)
VIOLATION(Predicate 4_22): Internal File Message_Queue.inl is included by Message_Queue.top from Framework(1) — Component(1,4)
VIOLATION(Predicate 4_22): Internal File Module.inl is included by Module.top from Framework(1) — Component(1,5)
VIOLATION(Predicate 4_22): Internal File Multiplexor.inl is included by Multiplexor.top from Framework(1) — Component(1,6)
VIOLATION(Predicate 4_22): Internal File Multiplexor.inl is included by Stream_Modules.top from Framework(1) — Component(1,8)
VIOLATION(Predicate 4_22): Internal File Stream.inl is included by Stream.top from Framework(1) — Component(1,7)
VIOLATION(Predicate 4_22): Internal File Task.inl is included by Task.top from Framework(1) — Component(1,9)
VIOLATION(Predicate 4_22): Internal File CORBA_Handler.inl is included by CORBA_Handler.top from Framework(2) — Component(2,1)
VIOLATION(Predicate 4_22): Internal File CORBA_Ref.inl is included by CORBA_Ref.top from Framework(2) — Component(2,2)
VIOLATION(Predicate 4_22): Internal File Set.inl is included by Set.top from Framework(3) — Component(3,1)
VIOLATION(Predicate 4_22): Internal File Stack.inl is included by Stack.top from Framework(3) — Component(3,2)
VIOLATION(Predicate 4_22): Internal File SString.inl is included by SString.top from Framework(3) — Component(3,3)
VIOLATION(Predicate 4_22): Internal File Process_Manager.inl is included by Process_Manager.top from Framework(4) — Component(4,4)
VIOLATION(Predicate 4_22): Internal File Synch.inl is included by Synch.top from Framework(4) — Component(4,5)
VIOLATION(Predicate 4_22): Internal File Synch_T.inl is included by Synch_T.top from Framework(4) — Component(4,7)
VIOLATION(Predicate 4_22): Internal File Thread.inl is included by Thread.top from Framework(4) — Component(4,8)

VIOLATION(Predicate 4_22): Internal File Thread_Manager.inl is included by Thread_Manager.top from Framework(4) — Component(4,9)
VIOLATION(Predicate 4_22): Internal File Token.inl is included by Token.top from Framework(4) — Component(4,10)
VIOLATION(Predicate 4_22): Internal File Svc_Handler.inl is included by Svc_Handler.top from Framework(5) — Component(5,5)
VIOLATION(Predicate 4_22): Internal File ACE.inl is included by Log_Msg.top from Framework(7) — Component(7,1)
VIOLATION(Predicate 4_22): Internal File ACE.inl is included by Time_Value.inm from Framework(11) — Component(11,8)
VIOLATION(Predicate 4_22): Internal File Event_Handler.inl is included by Event_Handler.top from Framework(11) — Component(11,1)
VIOLATION(Predicate 4_22): Internal File Event_Handler_T.inl is included by Event_Handler_T.top from Framework(11) — Component(11,2)
VIOLATION(Predicate 4_22): Internal File Handle_Set.inl is included by Handle_Set.top from Framework(11) — Component(11,3)
VIOLATION(Predicate 4_22): Internal File Proactor.inl is included by Proactor.top from Framework(11) — Component(11,4)
VIOLATION(Predicate 4_22): Internal File ReactorEx.inl is included by ReactorEx.top from Framework(11) — Component(11,6)
VIOLATION(Predicate 4_22): Internal File Signal.inl is included by Signal.top from Framework(11) — Component(11,7)
VIOLATION(Predicate 4_22): Internal File Time_Value.inl is included by Time_Value.top from Framework(11) — Component(11,8)
VIOLATION(Predicate 4_22): Internal File Timer_Queue.inl is included by Time_Value.inm from Framework(11) — Component(11,8)
VIOLATION(Predicate 4_22): Internal File Timer_Queue.inl is included by Timer_Queue.top from Framework(11) — Component(11,9)
Checking for predicate 4_22 is complete —— STATUS:false
Checking for Predicate 4_15 is complete —— STATUS: false
==========================================================
Checking for Predicate 4_11 is complete —— STATUS: false
==========================================================
Checking for Predicate 4_12: InclusionLoopsMayNotExist
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    4 |—> 5 |—> 4
    DETAILS:
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
        NodeNumber(5)   FileName(Map_Manager.inm)   Framework(1) Component(1,2)
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    5 |—> 4 |—> 5
    DETAILS:
        NodeNumber(5)   FileName(Map_Manager.inm)   Framework(1) Component(1,2)
        NodeNumber(4)   FileName(Map_Manager.top) Framework(1) Component(1,2)
        NodeNumber(5)   FileName(Map_Manager.inm)   Framework(1) Component(1,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    10 |—> 11 |—> 10
    DETAILS:
        NodeNumber(10)   FileName(Message_Queue.top)   Framework(1) Component(1,4)
        NodeNumber(11)   FileName(Message_Queue.inm)   Framework(1) Component(1,4)
        NodeNumber(10)   FileName(Message_Queue.top)   Framework(1) Component(1,4)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    11 |—> 10 |—> 11
    DETAILS:
        NodeNumber(11)   FileName(Message_Queue.inm)   Framework(1) Component(1,4)
        NodeNumber(10)   FileName(Message_Queue.top)   Framework(1) Component(1,4)
        NodeNumber(11)   FileName(Message_Queue.inm)   Framework(1) Component(1,4)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    13 |—> 14 |—> 13
    DETAILS:
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    13 |—> 23 |—> 22 |—> 17 |—> 14 |—> 13
    DETAILS:
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
        NodeNumber(23)   FileName(Stream_Modules.inm)   Framework(1) Component(1,8)
        NodeNumber(22)   FileName(Stream_Modules.top)   Framework(1) Component(1,8)
        NodeNumber(17)   FileName(Multiplexor.inm)   Framework(1) Component(1,6)
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    14 |—> 13 |—> 14
    DETAILS:
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    14 |—> 13 |—> 23 |—> 22 |—> 17 |—> 14
    DETAILS:
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)
        NodeNumber(13)   FileName(Module.top)   Framework(1) Component(1,5)
        NodeNumber(23)   FileName(Stream_Modules.inm)   Framework(1) Component(1,8)
        NodeNumber(22)   FileName(Stream_Modules.top)   Framework(1) Component(1,8)
        NodeNumber(17)   FileName(Multiplexor.inm)   Framework(1) Component(1,6)
        NodeNumber(14)   FileName(Module.inm)   Framework(1) Component(1,5)

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    17 |--> 14 |--> 13 |--> 23 |--> 22 |--> 17
    DETAILS:
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    19 |--> 20 |--> 19
    DETAILS:
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    20 |--> 19 |--> 20
    DETAILS:
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
        NodeNumber(19)  FileName(Stream.top)  Framework(1) Component(1,7)
        NodeNumber(20)  FileName(Stream.inm)  Framework(1) Component(1,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    22 |--> 17 |--> 14 |--> 13 |--> 23 |--> 22
    DETAILS:
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    23 |--> 22 |--> 17 |--> 14 |--> 13 |--> 23
    DETAILS:
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
        NodeNumber(22)  FileName(Stream_Modules.top)  Framework(1) Component(1,8)
        NodeNumber(17)  FileName(Multiplexor.inm)  Framework(1) Component(1,6)
        NodeNumber(14)  FileName(Module.inm)  Framework(1) Component(1,5)
        NodeNumber(13)  FileName(Module.top)  Framework(1) Component(1,5)
        NodeNumber(23)  FileName(Stream_Modules.inm)  Framework(1) Component(1,8)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    31 |--> 32 |--> 31
    DETAILS:
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    32 |--> 31 |--> 32
    DETAILS:
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
        NodeNumber(31)  FileName(CORBA_Ref.top)  Framework(2) Component(2,2)
        NodeNumber(32)  FileName(CORBA_Ref.inm)  Framework(2) Component(2,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    34 |--> 35 |--> 34
    DETAILS:
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    35 |--> 34 |--> 35
    DETAILS:
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
        NodeNumber(34)  FileName(Set.top)  Framework(3) Component(3,1)
        NodeNumber(35)  FileName(Set.inm)  Framework(3) Component(3,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    37 |--> 38 |--> 37
    DETAILS:
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    38 |--> 37 |--> 38
    DETAILS:
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
        NodeNumber(37)  FileName(Stack.top)  Framework(3) Component(3,2)
        NodeNumber(38)  FileName(Stack.inm)  Framework(3) Component(3,2)
```

```
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    46 |—> 47 |—> 46
    DETAILS:
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    47 |—> 46 |—> 47
    DETAILS:
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
        NodeNumber(46)  FileName(Future.top)  Framework(4) Component(4,2)
        NodeNumber(47)  FileName(Future.inm)  Framework(4) Component(4,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    56 |—> 62 |—> 56
    DETAILS:
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    56 |—> 62 |—> 61 |—> 143 |—> 56
    DETAILS:
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)  Framework(11)    Component(11,8)
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    61 |—> 62 |—> 61
    DETAILS:
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    61 |—> 143 |—> 56 |—> 62 |—> 61
    DETAILS:
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)  Framework(11)    Component(11,8)
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 56 |—> 62
    DETAILS:
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 61 |—> 62
    DETAILS:
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    62 |—> 61 |—> 143 |—> 56 |—> 62
    DETAILS:
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
        NodeNumber(61)  FileName(Synch_T.top)  Framework(4) Component(4,7)
        NodeNumber(143) FileName(Time_Value.inm)  Framework(11)    Component(11,8)
        NodeNumber(56)  FileName(Synch.inm)  Framework(4) Component(4,5)
        NodeNumber(62)  FileName(Synch_T.inm) Framework(4) Component(4,7)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    73 |—> 74 |—> 73
    DETAILS:
        NodeNumber(73)  FileName(Acceptor.top)  Framework(5) Component(5,1)
        NodeNumber(74)  FileName(Acceptor.inm) Framework(5) Component(5,1)
        NodeNumber(73)  FileName(Acceptor.top)  Framework(5) Component(5,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    74 |—> 73 |—> 74
    DETAILS:
        NodeNumber(74)  FileName(Acceptor.inm) Framework(5) Component(5,1)
        NodeNumber(73)  FileName(Acceptor.top)  Framework(5) Component(5,1)
        NodeNumber(74)  FileName(Acceptor.inm) Framework(5) Component(5,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
    76 |—> 77 |—> 76
    DETAILS:
```

NodeNumber(76) FileName(Connector.top) Framework(5) Component(5,2)
NodeNumber(77) FileName(Connector.inm) Framework(5) Component(5,2)
NodeNumber(76) FileName(Connector.top) Framework(5) Component(5,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
77 |—> 76 |—> 77
DETAILS:
NodeNumber(77) FileName(Connector.inm) Framework(5) Component(5,2)
NodeNumber(76) FileName(Connector.top) Framework(5) Component(5,2)
NodeNumber(77) FileName(Connector.inm) Framework(5) Component(5,2)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
79 |—> 80 |—> 79
DETAILS:
NodeNumber(79) FileName(Dynamic_Service.top) Framework(5) Component(5,3)
NodeNumber(80) FileName(Dynamic_Service.inm) Framework(5) Component(5,3)
NodeNumber(79) FileName(Dynamic_Service.top) Framework(5) Component(5,3)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
80 |—> 79 |—> 80
DETAILS:
NodeNumber(80) FileName(Dynamic_Service.inm) Framework(5) Component(5,3)
NodeNumber(79) FileName(Dynamic_Service.top) Framework(5) Component(5,3)
NodeNumber(80) FileName(Dynamic_Service.inm) Framework(5) Component(5,3)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
85 |—> 86 |—> 85
DETAILS:
NodeNumber(85) FileName(Svc_Handler.top) Framework(5) Component(5,5)
NodeNumber(86) FileName(Svc_Handler.inm) Framework(5) Component(5,5)
NodeNumber(85) FileName(Svc_Handler.top) Framework(5) Component(5,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
86 |—> 85 |—> 86
DETAILS:
NodeNumber(86) FileName(Svc_Handler.inm) Framework(5) Component(5,5)
NodeNumber(85) FileName(Svc_Handler.top) Framework(5) Component(5,5)
NodeNumber(86) FileName(Svc_Handler.inm) Framework(5) Component(5,5)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
122 |—> 123 |—> 122
DETAILS:
NodeNumber(122) FileName(Event_Handler.inm) Framework(11) Component(11,1)
NodeNumber(123) FileName(Event_Handler.inl) Framework(11) Component(11,1)
NodeNumber(122) FileName(Event_Handler.inm) Framework(11) Component(11,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
123 |—> 122 |—> 123
DETAILS:
NodeNumber(123) FileName(Event_Handler.inl) Framework(11) Component(11,1)
NodeNumber(122) FileName(Event_Handler.inm) Framework(11) Component(11,1)
NodeNumber(123) FileName(Event_Handler.inl) Framework(11) Component(11,1)
VIOLATION(Predicate4_13): FileInclusionLOOP has been detected:
143 |—> 56 |—> 62 |—> 61 |—> 143
DETAILS:
NodeNumber(143) FileName(Time_Value.inm) Framework(11) Component(11,8)
NodeNumber(56) FileName(Synch.inm) Framework(4) Component(4,5)
NodeNumber(62) FileName(Synch_T.inm) Framework(4) Component(4,7)
NodeNumber(61) FileName(Synch_T.top) Framework(4) Component(4,7)
NodeNumber(143) FileName(Time_Value.inm) Framework(11) Component(11,8)
Checking for Predicate 4_12 is complete —— STATUS: false

Checking for Predicate 4_8 is complete —— STATUS: false

Checking for GoodFileStructure_predicate4_5 is complete —— STATUS:false

# APPENDIX B

# A PROTOTYPE FOR CO–TOOLKITS' DESIGN AND IMPLEMENTATION STRUCTURES

# Appendix B1

# A Prototype For CO–Toolkits' Design and Implementation Structures: An Overview

## B1.1. Overview

Similar to the previous prototype (*c.f. Appendix A*), Java has been used to build this prototype based on the view for CO–Toolkits' design and implementation structures (*c.f. chapter 4*). The prototype's data structure represents the view structure of section 5.2.2. Each predicate from the view's characteristics section (*c.f. section 5.2.3*) is implemented as method that returns a boolean value (i.e. TRUE or FALSE). For convenience, we named each of these methods by its formula number as in chapter 5.

Similar to the previous prototype, we kept this prototype simple. All of its inputs are built–in. Currently, we have three built–in examples on design and implementation, one of them is on a sample ACE inheritance structure.

To run this prototype:

(1) The twenty two files listed in the next section should be in a directory named **V2_DesignAndImplementation_R3**. In Java, path and file names are case sensitive.

(2) Run the command
**javac –deprecation V2_DesignAndImplementation_R3/*.java**

(3) Run the command

**java V2_DesignAndImplementation_R3.CheckDesignAndImplementation N**

whare **N** is the argument. It represents the example number of choice (i.e. 1 for example 1, 2 for example 2, and 3 for the ACE example).

The prototype operation goes through the following steps:

(1) It assigns values for the various constants such as ***F*** (i.e. number of frameworks), *S* (i.e. number of shared pools), ***C[]*** (i.e. numbers of components in each framework), ***c[][]*** (numbers of classes in each component), ***MP[][][]*** (i.e. number of public methods in each class), ***MR[][][]*** (i.e. number of protected methods in each class), ***MV[][][]*** (i.e. number of private methods in each class), ***OP[][][]*** (i.e. number of public objects in each class), and etc.

(2) It allocates its internal data structure based on the view structure in section 5.2.2.

(3) It updates its internal data structures with the classes names, methods names, objects names, and inheritance relations.

(4) It checks its internal data structure by sequentially invoking its methods that represent the various predicate (i.e. rules) in the view characteristic section (*c.f. section 5.2.3*). Each of these methods reports any violation to its embedded rule.

# B1.2. Source Code Description

This section describes this prototype's source files:

## B1.2.1. CheckDesignAndImplementation.java

This file contains the implementation of all predicates that represent the characteristics of the Design and Implementation's View. These predicates are named by their formula numbers as in chapter 5. For example, predicate5–40 is the implementation of characteristic (5–40). Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: CheckDesignAndImplementation.java
 */
public class CheckDesignAndImplementation_R3 {
  private static String output;public CheckDesignAndImplementation_R3() {
  super();
}
private static boolean Associates_Bi_ClassLevel(int fr_id1, int co_id1, int cl_id1,int fr_id2, int co_id2, int
 cl_id2){
  boolean returnValue1=One_to_One_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
 fr_id2, co_id2, cl_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,fr_id2,
 co_id2, cl_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,fr_id2,
 co_id2, cl_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,fr_id2,
 co_id2, cl_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,fr_id2,
 co_id2, cl_id2);
  boolean returnValue=((!returnValue1)||(returnValue2)||(!returnValue3)||(returnValue4)||(returnValue5));
  return returnValue;
}
private static boolean Associates_Bi_SharedClassLevel(int shp_id1, int shc_id1,int shp_id2, int shc_id2){
  boolean
returnValue1=One_to_One_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue2=One_to_One_BiDirectionalAssociation_SharedCLassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue3=One_to_M_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue4=One_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
  returnValue5=M_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
  returnValue= ((!returnValue1)||(returnValue2)||(!returnValue3)||(returnValue4)||(returnValue5));
  return returnValue;
}
private static boolean Associates_ClassLevel(int fr_id1, int co_id1, int cl_id1,int fr_id2, int co_id2, int cl_id2){
  boolean returnValue1=One_to_One_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                                              fr_id2,co_id2, cl_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                                              fr_id2, co_id2, cl_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                                              fr_id2, co_id2, cl_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                                                fr_id2, co_id2, cl_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                                                fr_id2, co_id2, cl_id2);
  boolean returnValue= returnValue1||returnValue2||returnValue3||returnValue4||returnValue5;
```

```
  return returnValue;
}
private static boolean Associates_ComponentLevel(int fr_id1, int co_id1,int fr_id2, int co_id2){
  boolean returnValue1=One_to_One_UniDirectionalAssociation_ComponentLevel(fr_id1,
                                                                        co_id1, fr_id2, co_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_ComponentLevel(fr_id1, co_id1, fr_id2,
                                                                        co_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_ComponentLevel(fr_id1, co_id1, fr_id2,
                                                                        co_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_ComponentLevel(fr_id1,
                                                                        co_id1, fr_id2, co_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_ComponentLevel(fr_id1, co_id1, fr_id2, co_id2);
  boolean returnValue= returnValue1||returnValue2||returnValue3||returnValue4||returnValue5;
  return returnValue;
}
private static boolean Associates_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue1=One_to_One_UniDirectionalAssociation_FrameworkLevel(fr_id1, fr_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_FrameworkLevel(fr_id1, fr_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_FrameworkLevel(fr_id1, fr_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_FrameworkLevel(fr_id1, fr_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_FrameworkLevel(fr_id1, fr_id2);
  boolean returnValue= returnValue1||returnValue2||returnValue3||returnValue4||returnValue5;
  return returnValue;
}
private static boolean Associates_SharedClassLevel(int shp_id1, int shc_id1,int shp_id2, int shc_id2){
  boolean
returnValue1=One_to_One_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue2=One_to_One_BiDirectionalAssociation_SharedCLassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue3=One_to_M_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue4=One_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
 returnValue5=M_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean returnValue= returnValue1||returnValue2||returnValue3||returnValue4||returnValue5;
  return returnValue;
}
private static boolean Associates_SharedPoolLevel(int shp_id1,int shp_id2){
  boolean returnValue1=One_to_One_UniDirectionalAssociation_SharedPoolLevel(shp_id1, shp_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_SharedPoolLevel(shp_id1, shp_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_SharedPoolLevel(shp_id1, shp_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_SharedPoolLevel(shp_id1, shp_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_SharedPoolLevel(shp_id1, shp_id2);
  boolean returnValue= returnValue1||returnValue2||returnValue3||returnValue4||returnValue5;
  return returnValue;
}
private static boolean Associates_Uni_ClassLevel(int fr_id1, int co_id1, int cl_id1,int fr_id2, int co_id2, int
cl_id2){
  boolean
 returnValue1=One_to_One_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
        fr_id2, co_id2, cl_id2);
  boolean returnValue2=One_to_One_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                 fr_id2, co_id2, cl_id2);
  boolean returnValue3=One_to_M_UniDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                 fr_id2, co_id2, cl_id2);
  boolean returnValue4=One_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
```

```
                                   fr_id2, co_id2, cl_id2);
  boolean returnValue5=M_to_M_BiDirectionalAssociation_ClassLevel(fr_id1,co_id1,cl_id1,
                                   fr_id2, co_id2, cl_id2);
  boolean returnValue=((returnValue1)||(!returnValue2)||(returnValue3)||(!returnValue4)||(!returnValue5));
  return returnValue;
}
private static boolean Associates_Uni_SharedClassLevel(int shp_id1, int shc_id1,int shp_id2, int shc_id2){
boolean
returnValue1=One_to_One_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
returnValue2=One_to_One_BiDirectionalAssociation_SharedCLassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
returnValue3=One_to_M_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
returnValue4=One_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean
returnValue5=M_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
  boolean returnValue= ((returnValue1)||(!returnValue2)||(returnValue3)||(!returnValue4)||(!returnValue5));
  return returnValue;
}
private static void displayDesignAndImplementationStructure() {}
private static void eXample_1() {DesignAndImplementationStructure.initializeFor_Example_1();}
private static void eXample_2() {DesignAndImplementationStructure.initializeFor_Example_2();}
private static void eXample_3_ACE() {DesignAndImplementationStructure.initializeFor_Example_3_ACE();
}
private static boolean Inherits_ClassLevel(int fr_id1, int co_id1,int cl_id1,
                              int fr_id2, int co_id2, int cl_id2){
  boolean returnValue=false;
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Publi-
cInheritedClasses[ii]!=null){
      int inh_fr_id=
   DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PublicInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
      int inh_co_id=
   DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PublicInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
      int inh_cl_id=
   DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PublicInherited-
Classes[ii].ClassNodeNumber;
      if ((inh_fr_id==fr_id2)&&(inh_co_id==co_id2)&&(inh_cl_id==cl_id2)){
        returnValue=true;
        break;
      }
      ii=ii+1;
    }
  }
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
  (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pro-
tectedInheritedClasses[ii]!=null){
      int inh_fr_id=
```

```
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ProtectedInheri-
tedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
      int inh_co_id=
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ProtectedInheri-
tedClasses[ii].ParentComponent.ComponentNodeNumber;
      int inh_cl_id=
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ProtectedInheri-
tedClasses[ii].ClassNodeNumber;
      if ((inh_fr_id==fr_id2)&&(inh_co_id==co_id2)&&(inh_cl_id==cl_id2)){
        returnValue=true;
        reportViolation("Class("+fr_id1+","+co_id1+","+cl_id1+")   inherits   in   protected   from
Class("+fr_id2+","+co_id2+","+cl_id2+")");
        break;
      }
      ii=ii+1;
    }
  }
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
  (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    while  (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Priva-
teInheritedClasses[ii]!=null){
      int inh_fr_id=
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PrivateInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
      int inh_co_id=
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
      int inh_cl_id=
DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PrivateInherited-
Classes[ii].ClassNodeNumber;
      if ((inh_fr_id==fr_id2)&&(inh_co_id==co_id2)&&(inh_cl_id==cl_id2)){
        returnValue=true;
        reportViolation("Class("+fr_id1+","+co_id1+","+cl_id1+")   inherits   in   private   from
Class("+fr_id2+","+co_id2+","+cl_id2+")");
        break;
      }
      ii=ii+1;
    }
  }
  return returnValue;
}
/**
 * @return boolean
 */
private static boolean Inherits_SharedClassLevel(int shp_id1,int shc_id1,int shp_id2,int shc_id2){
  boolean returnValue=false;
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
    (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    while  (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicInheritedShared-
Classes[ii]!=null){
      int inh_shp_id=
            DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber;
      int inh_shc_id=
```

```
            DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicInheritedShared-
Classes[ii].ClassNodeNumber;
      if ((inh_shp_id==shp_id2)&&(inh_shc_id==shc_id2)){
        returnValue=true;
        break;
      }
      ii=ii+1;
    }
  }
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
  (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedInheritedSha-
redClasses[ii]!=null){
      int inh_shp_id=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedIn-
heritedSharedClasses[ii].ParentSharedPoolNodeNumber;
      int inh_shc_id=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedIn-
heritedSharedClasses[ii].ClassNodeNumber;
      if ((inh_shp_id==shp_id2)&&(inh_shc_id==shc_id2)){
        returnValue=true;
        reportViolation("SharedClass("+shp_id1+","+shc_id1+") inherits in protected from Shared-
Class("+shp_id2+","+shc_id2+")");
        break;
      }
      ii=ii+1;
    }
  }
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
  (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateInheritedShared-
Classes[ii]!=null){
      int inh_shp_id=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateInheri-
tedSharedClasses[ii].ParentSharedPoolNodeNumber;
      int inh_shc_id=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateInheri-
tedSharedClasses[ii].ClassNodeNumber;
      if ((inh_shp_id==shp_id2)&&(inh_shc_id==shc_id2)){
        returnValue=true;
        reportViolation("SharedClass("+shp_id1+","+shc_id1+") inherits in private from Shared-
Class("+shp_id2+","+shc_id2+")");
        break;
      }
      ii=ii+1;
    }
  }
  return returnValue;
}
public static void initializeDesignAndImplementationStructure() {
if (Globals.EXAMPLE_NOM==1){
    eXample_1();
  }else{
    if (Globals.EXAMPLE_NOM==2){
      eXample_2();
    }else{
      if (Globals.EXAMPLE_NOM==3){
        eXample_3_ACE();
```

```
        }
        else{ System.out.println("Wrong Example Nomber.It has to be 1, 2, or 3");
          return;
        }
      }
    }// eXample_1();}
  /**
   * @return boolean
   */
  // need to determine what parameters to pass
  private static boolean M_to_M_BiDirectionalAssociation_ClassLevel(
                        int fr_id1, int co_id1,  int cl_id1,
                        int fr_id2, int co_id2,  int cl_id2){
    boolean returnValue=true;
    if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
      (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
      int ii=1;
      int cnt=0;
      String                    className=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ID.ClassName;
      while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pu-
blicSection.Object[ii]!=null){
        String                  objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PublicSection.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      ii=1;
      while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pro-
tectedSection.Object[ii]!=null){
        String                  objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      ii=1;
      while   (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Priva-
teSection.Object[ii]!=null){
        String                  objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      if (cnt>1){
        ii=1;
        int cnt2=0;
        className=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ID.ClassName;
        while     (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pu-
blicSection.Object[ii]!=null){
          String                objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PublicSection.ObjectType[ii];
```

```
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      ii=1;
      while    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pro-
tectedSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      ii=1;
      while    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pri-
vateSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      if (cnt2<2){returnValue=false;}
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
/**
 * @return boolean
 */
// need to determine what parameters to pass
private static boolean M_to_M_BiDirectionalAssociation_ComponentLevel(
                        int fr_id1, int co_id1,
                        int fr_id2, int co_id2){ boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[ii]!=null){
    int                        cl_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[ii].ID.ClassNodeNumber;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[jj]!=null){
      int                      cl_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=M_to_M_BiDirectionalAssociation_ClassLe-
vel(fr_id1,co_id1,cl_id1,fr_id2,co_id2,cl_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
/**
```

```
 * @return boolean
 */
// need to determine what parameters to pass
private static boolean M_to_M_BiDirectionalAssociation_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii]!=null){
    int    co_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii].ID.ComponentNode-
Number;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj]!=null){
      int    co_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj].ID.ComponentNo-
deNumber;
      returnValue=M_to_M_BiDirectionalAssociation_ComponentLevel(fr_id1,co_id1,fr_id2,co_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean M_to_M_BiDirectionalAssociation_SharedClassLevel(
                           int shp_id1, int shc_id1,
                           int shp_id2, int shc_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
    (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    String className=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ClassName;
    while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSection.Ob-
ject[ii]!=null){
      String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while      (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedSection.Ob-
ject[ii]!=null){
      String           objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].Protec-
tedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSection.Ob-
ject[ii]!=null){
      String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
```

```
      ii=ii+1;
    }
    if (cnt>1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ClassName;
      while    (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSection.Ob-
ject[ii]!=null){
        String   objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      ii=1;
      while   (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.Ob-
ject[ii]!=null){
        String       objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].Protec-
tedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      ii=1;
      while      (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.Ob-
ject[ii]!=null){
        String   objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
        }
        ii=ii+1;
      }
      if (cnt2<2){returnValue=false;}
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;}
/**
 * @return boolean
 */
// need to determine what parameters to pass
private static boolean M_to_M_BiDirectionalAssociation_SharedPoolLevel(int shp_id1, int shp_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii]!=null){
    int   shc_id1=DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii].ID.ParentSharedPoolNo-
deNumber;
    int jj=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj]!=null){
      int shc_id2=DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=M_to_M_BiDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
```

```
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
 }
 public static void main(String args[]) { Globals.EXAMPLE_NOM= Integer.parseInt(args[0]);
  initializeDesignAndImplementationStructure();
  predicate5_18_GoodDesignAndImplementation();
 }
 private static boolean One_to_M_BiDirectionalAssociation_ClassLevel(
                    int fr_id1, int co_id1,  int cl_id1,
                    int fr_id2, int co_id2,  int cl_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    String                    className=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ID.ClassName;
    while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pu-
blicSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PublicSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pro-
tectedSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ProtectedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while    (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Priva-
teSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PrivateSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    if (cnt>1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ID.ClassName;
      while     (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pu-
blicSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PublicSection.ObjectType[ii];
        if (objType==className){
```

```
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while  (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pro-
tectedSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while  (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pri-
vateSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_M_BiDirectionalAssociation_ComponentLevel(
                              int fr_id1, int co_id1,
                              int fr_id2, int co_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[ii]!=null){
    int                    cl_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[ii].ID.ClassNodeNumber;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[jj]!=null){
      int                    cl_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_M_BiDirectionalAssociation_ClassLe-
vel(fr_id1,co_id1,cl_id1,fr_id2,co_id2,cl_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_M_BiDirectionalAssociation_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue=true;
  int ii=1;
```

```
    while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii]!=null){
      int  co_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii].ID.ComponentNode-
  Number;
      int jj=1;
      while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj]!=null){
        int  co_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj].ID.ComponentNo-
  deNumber;
        returnValue=One_to_M_BiDirectionalAssociation_ComponentLevel(fr_id1,co_id1,fr_id2,co_id2);
        if (!returnValue){break;}
        jj=jj+1;
      }
      if (!returnValue) {break;}
      ii=ii+1;
    }
    return returnValue;
  }
  private static boolean One_to_M_BiDirectionalAssociation_SharedClassLevel(
                                  int shp_id1, int shc_id1,
                                  int shp_id2, int shc_id2){
    boolean returnValue=true;
    if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
      (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
      int ii=1;
      int cnt=0;
      String className=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ClassName;
      while       (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSection.Ob-
  ject[ii]!=null){
        String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSec-
  tion.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      ii=1;
      while     (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedSection.Ob-
  ject[ii]!=null){
        String        objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].Protec-
  tedSection.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      ii=1;
      while       (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSection.Ob-
  ject[ii]!=null){
        String    objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSec-
  tion.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
        }
        ii=ii+1;
      }
      if (cnt>1){
        ii=1;
        int cnt2=0;
        className=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ClassName;
```

```
      while      (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSection.Ob-
ject[ii]!=null){
        String   objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while   (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.Ob-
ject[ii]!=null){
        String        objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].Protec-
tedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while      (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.Ob-
ject[ii]!=null){
        String  objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_M_BiDirectionalAssociation_SharedPoolLevel(int shp_id1, int shp_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii]!=null){
    int   shc_id1=DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii].ID.ParentSharedPoolNo-
deNumber;
    int jj=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj]!=null){
      int shc_id2=DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_M_BiDirectionalAssociation_SharedClassLe-
vel(shp_id1,shc_id1,shp_id2,shc_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
```

```
private static boolean One_to_M_UniDirectionalAssociation_ClassLevel(
                        int fr_id1, int co_id1,  int cl_id1,
                        int fr_id2, int co_id2,  int cl_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    String                    className=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ID.ClassName;
    while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pu-
blicSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PublicSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while     (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pro-
tectedSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ProtectedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while   (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Priva-
teSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PrivateSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    if (cnt>1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ID.ClassName;
      while     (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pu-
blicSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PublicSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while   (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pro-
tectedSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
```

```
nent[co_id2].Class[cl_id2].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pri-
vateSection.Object[ii]!=null){
        String                    objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_M_UniDirectionalAssociation_ComponentLevel(
                              int fr_id1, int co_id1,
                              int fr_id2, int co_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[ii]!=null){
    int                      cl_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[ii].ID.ClassNodeNumber;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[jj]!=null){
      int                    cl_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_M_UniDirectionalAssociation_ClassLe-
vel(fr_id1,co_id1,cl_id1,fr_id2,co_id2,cl_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_M_UniDirectionalAssociation_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii]!=null){
    int    co_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii].ID.ComponentNode-
Number;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj]!=null){
      int    co_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj].ID.ComponentNo-
deNumber;
      returnValue=One_to_M_UniDirectionalAssociation_ComponentLevel(fr_id1,co_id1,fr_id2,co_id2);
      if (!returnValue){break;}
```

```
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_M_UniDirectionalAssociation_SharedClassLevel(
                              int shp_id1, int shc_id1,
                              int shp_id2, int shc_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
    (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    String className=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ClassName;
    while          (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSection.Ob-
ject[ii]!=null){
      String    objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while      (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedSection.Ob-
ject[ii]!=null){
      String          objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].Protec-
tedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    ii=1;
    while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSection.Ob-
ject[ii]!=null){
      String    objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
      }
      ii=ii+1;
    }
    if (cnt>1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ClassName;
      while        (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSection.Ob-
ject[ii]!=null){
        String   objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
```

```
    }
    ii=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.Object[ii]!=null){
      String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.ObjectType[ii];
      if (objType==className){
        cnt2=cnt2+1;
        if (cnt2>=1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.Object[ii]!=null){
      String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.ObjectType[ii];
      if (objType==className){
        cnt2=cnt2+1;
        if (cnt2>=1){returnValue=false; break; }
      }
      ii=ii+1;
    }
  }
  else{returnValue=false; }
 }
 else{returnValue=false;}
 return returnValue;
}
private static boolean One_to_M_UniDirectionalAssociation_SharedPoolLevel(int shp_id1,int shp_id2){
 boolean returnValue=true;
 int ii=1;
 while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii]!=null){
   int shc_id1=DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii].ID.ParentSharedPoolNodeNumber;
   int jj=1;
   while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj]!=null){
     int shc_id2=DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj].ID.ClassNodeNumber;
     returnValue=One_to_M_UniDirectionalAssociation_SharedClassLevel(shp_id1,shc_id1,shp_id2,shc_id2);
     if (!returnValue){break;}
     jj=jj+1;
   }
   if (!returnValue) {break;}
   ii=ii+1;
 }
 return returnValue;
}
private static boolean One_to_One_BiDirectionalAssociation_ClassLevel(
                    int fr_id1, int co_id1,  int cl_id1,
                    int fr_id2, int co_id2,  int cl_id2){
 boolean returnValue=true;
 if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
   (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
   int ii=1;
   int cnt=0;
   String className=DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].ID.ClassName;
```

```
    while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PublicSection.Object[ii]!=null){
      String objType=DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PublicSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while ((cnt<=1) && (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ProtectedSection.Object[ii]!=null)){
      String objType=DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ProtectedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while ((cnt<=1) && (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PrivateSection.Object[ii]!=null)){
      String objType=DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].PrivateSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    if (cnt==1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].ID.ClassName;
      while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].PublicSection.Object[ii]!=null){
        String objType=DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].PublicSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while ((cnt2<=1) && (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].ProtectedSection.Object[ii]!=null)){
        String objType=DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
```

```
      ii=1;
      while ((cnt2<=1) && (DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.Object[ii]!=null)){
        String objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false;}
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_One_BiDirectionalAssociation_ComponentLevel(
                        int fr_id1, int co_id1,
                        int fr_id2, int co_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[ii]!=null){
    int cl_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[ii].ID.ClassNodeNumber;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[jj]!=null){
      int cl_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_One_BiDirectionalAssociation_ClassLe-
vel(fr_id1,co_id1,cl_id1,fr_id2,co_id2,cl_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_One_BiDirectionalAssociation_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii]!=null){
    int co_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii].ID.ComponentNode-
Number;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj]!=null){
      int co_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj].ID.ComponentNo-
deNumber;
      returnValue=One_to_One_BiDirectionalAssociation_ComponentLevel(fr_id1,co_id1,fr_id2,co_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
```

```
    return returnValue;
  }
  private static boolean One_to_One_BiDirectionalAssociation_SharedCLassLevel(
                                int shp_id1, int shc_id1,
                                int shp_id2, int shc_id2){
    boolean returnValue=true;
    if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
      (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
      int ii=1;
      int cnt=0;
      String className=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ClassName;
      while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSection.Ob-
ject[ii]!=null){
        String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
          if (cnt>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while      (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedSection.Ob-
ject[ii]!=null){
        String          objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].Protec-
tedSection.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
          if (cnt>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSection.Ob-
ject[ii]!=null){
        String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt=cnt+1;
          if (cnt>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      if (cnt==1){
        ii=1;
        int cnt2=0;
        className=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ClassName;
        while        (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSection.Ob-
ject[ii]!=null){
          String   objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSec-
tion.ObjectType[ii];
          if (objType==className){
            cnt2=cnt2+1;
            if (cnt2>1){returnValue=false; break; }
          }
          ii=ii+1;
        }
        ii=1;
```

```
      while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.Ob-
ject[ii]!=null){
        String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].Protec-
tedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.Ob-
ject[ii]!=null){
        String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_One_BiDirectionalAssociation_SharedPoolLevel(int shp_id1,int shp_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii]!=null){
    int shc_id1=DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii].ID.ParentSharedPoolNo-
deNumber;
    int jj=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj]!=null){
      int shc_id2=DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_One_BiDirectionalAssociation_SharedCLassLe-
vel(shp_id1,shc_id1,shp_id2,shc_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_One_UniDirectionalAssociation_ClassLevel(
                    int fr_id1, int co_id1,  int cl_id1,
                    int fr_id2, int co_id2,  int cl_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1]!=null) &&
    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2]!=null)){
    int ii=1;
    int cnt=0;
    String className=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ID.ClassName;
    while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pu-
blicSection.Object[ii]!=null){
```

```
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PublicSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false;  break; }
      }
      ii=ii+1;
    }
    ii=1;
    while    (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Pro-
tectedSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ProtectedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while   (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[cl_id1].Priva-
teSection.Object[ii]!=null){
      String                    objType=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].PrivateSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    if (cnt==1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[cl_id1].ID.ClassName;
      while    (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[cl_id2].Pu-
blicSection.Object[ii]!=null){
        String                  objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PublicSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while      ((cnt2<=1)       &&       (DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ProtectedSection.Object[ii]!=null)){
        String                  objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].ProtectedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while      ((cnt2<=1)       &&       (DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
```

```
nent[co_id2].Class[cl_id2].PrivateSection.Object[ii]!=null)){
        String                          objType=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[cl_id2].PrivateSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
private static boolean One_to_One_UniDirectionalAssociation_ComponentLevel(
                              int fr_id1, int co_id1,
                              int fr_id2, int co_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[co_id1].Class[ii]!=null){
    int                         cl_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Compo-
nent[co_id1].Class[ii].ID.ClassNodeNumber;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[co_id2].Class[jj]!=null){
      int                         cl_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Compo-
nent[co_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_One_UniDirectionalAssociation_ClassLe-
vel(fr_id1,co_id1,cl_id1,fr_id2,co_id2,cl_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_One_UniDirectionalAssociation_FrameworkLevel(int fr_id1,int fr_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii]!=null){
    int   co_id1=DesignAndImplementationStructure.Frameworks[fr_id1].Component[ii].ID.ComponentNode-
Number;
    int jj=1;
    while (DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj]!=null){
      int   co_id2=DesignAndImplementationStructure.Frameworks[fr_id2].Component[jj].ID.ComponentNo-
deNumber;
      returnValue=One_to_One_UniDirectionalAssociation_ComponentLevel(fr_id1,co_id1,fr_id2,co_id2);
      if (!returnValue){break;}
      jj=jj+1;
    }
    if (!returnValue) {break;}
    ii=ii+1;
  }
  return returnValue;
}
```

```
private static boolean One_to_One_UniDirectionalAssociation_FromSharedPoolsToFrameworks(int fr_id, int
co_id, int cl_id,int shp_id,int shc_id){
  boolean returnValue=true;
  String                    classname=DesignAndImplementationStructure.Frameworks[fr_id].Compo-
nent[co_id].Class[cl_id].ID.ClassName;
  int ii=1;
  while          (DesignAndImplementationStructure.SharedPools[shp_id].Class[shc_id].PublicSection.Ob-
ject[ii]!=null){
    if          (DesignAndImplementationStructure.SharedPools[shp_id].Class[shc_id].PublicSection.Object-
Type[ii]==classname){
      returnValue=false;
      reportViolation("SharedClass("+shp_id+","+shc_id+")        contains        objects        from
Class("+fr_id+","+co_id+","+cl_id+")");
      break;
    }
    ii=ii+1;
  }
  return returnValue;
}
private static boolean One_to_One_UniDirectionalAssociation_SharedClassLevel(
                              int shp_id1, int shc_id1,
                              int shp_id2, int shc_id2){
  boolean returnValue=true;
  if((DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1]!=null) &&
    (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2]!=null)){
    int ii=1;
    int cnt=0;
    String className=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ClassName;
    while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSection.Ob-
ject[ii]!=null){
      String     objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PublicSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while      (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ProtectedSection.Ob-
ject[ii]!=null){
      String         objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].Protec-
tedSection.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
      ii=ii+1;
    }
    ii=1;
    while        (DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSection.Ob-
ject[ii]!=null){
      String    objType=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].PrivateSec-
tion.ObjectType[ii];
      if (objType==className){
        cnt=cnt+1;
        if (cnt>1){returnValue=false; break; }
      }
```

```
      ii=ii+1;
    }
    if (cnt==1){
      ii=1;
      int cnt2=0;
      className=DesignAndImplementationStructure.SharedPools[shp_id1].Class[shc_id1].ClassName;
      while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSection.Ob-
ject[ii]!=null){
        String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PublicSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].ProtectedSection.Ob-
ject[ii]!=null){
        String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].Protec-
tedSection.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
      ii=1;
      while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSection.Ob-
ject[ii]!=null){
        String objType=DesignAndImplementationStructure.SharedPools[shp_id2].Class[shc_id2].PrivateSec-
tion.ObjectType[ii];
        if (objType==className){
          cnt2=cnt2+1;
          if (cnt2>=1){returnValue=false; break; }
        }
        ii=ii+1;
      }
    }
    else{returnValue=false; }
  }
  else{returnValue=false;}
  return returnValue;
}
/**
 * @return boolean
 */
// need to determine what parameters to pass
private static boolean One_to_One_UniDirectionalAssociation_SharedPoolLevel(int shp_id1,int shp_id2){
  boolean returnValue=true;
  int ii=1;
  while (DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii]!=null){
    int shc_id1=DesignAndImplementationStructure.SharedPools[shp_id1].Class[ii].ID.ParentSharedPoolNo-
deNumber;
    int jj=1;
    while (DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj]!=null){
      int shc_id2=DesignAndImplementationStructure.SharedPools[shp_id2].Class[jj].ID.ClassNodeNumber;
      returnValue=One_to_One_UniDirectionalAssociation_SharedClassLe-
```

```
vel(shp_id1,shc_id1,shp_id2,shc_id2);
        if (!returnValue){break;}
        jj=jj+1;
      }
      if (!returnValue) {break;}
      ii=ii+1;
    }
    return returnValue;
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_18_GoodDesignAndImplementation( ) {
    reportViolation("Checking for Main predicate5_18: GoodDesignAndImplementation");
    boolean returnValue1=predicate5_19();
    boolean returnValue=returnValue1;
    reportViolation("Checking for Main predicate5_18: GoodDesignAndImplementation   is complete. STATUS:
"+returnValue);
    return (returnValue);
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_19(){
    reportViolation("Checking for Predicate5_19: GoodUseOfRelationsAmongComponents");
    boolean returnValue1=predicate5_20();
    boolean returnValue2=predicate5_21();
    boolean returnValue3=predicate5_22();
    boolean returnValue=returnValue1&&returnValue2&&returnValue3;
    reportViolation("Checking for Predicate5_19 is complete. STATUS: "+ returnValue);
    return returnValue;
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_20(){
    reportViolation("Checking for predicate5_20: SafeInheritance");
    boolean returnValue1=predicate5_23();
    boolean returnValue2=predicate5_24();
    boolean returnValue3=predicate5_25();
    boolean returnValue4=predicate5_26();
    boolean returnValue=returnValue1&&returnValue2&&returnValue3&&returnValue4;
    reportViolation("Checking for predicate5_20 is complete. STATUS: "+returnValue);
    return (returnValue);
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_21(){
    reportViolation("Checking for predicate5_21: GoodAssociation");
    boolean returnValue=true;
    if (Globals.EXAMPLE_NOM !=3) {
      boolean returnValue1=predicate5_27();
      boolean returnValue2=predicate5_28();
      returnValue=returnValue1&&returnValue2;
    }else{
      returnValue=true;
      // reportViolation("This Predicate has been disabled only for the ACE example 3");
```

```
  }
  reportViolation("Checking for predicate5_21 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_22(){
  reportViolation("Checking for predicate5_22: GoodUseOfOtherRelations");
  boolean returnValue=true;
  //This predicate is for future expansion purposes.
  reportViolation("Checking for predicate5_22 is complete. STATUS: "+returnValue);
  return (returnValue);
}
/**
 * @return boolean
 */
protected static boolean predicate5_23(){
  output="Checking for predicate 5_23: InheritanceMayNotExistAcrossFrameworks";
  boolean returnValue=true;
  for (int i=1; i<=DesignAndImplementationStructure.F;i++){
    for (int j=1;j<=DesignAndImplementationStructure.C[i];j++){
      for (int k=1; k<=DesignAndImplementationStructure.c[i][j]; k++){
        int ii=1;
        int fr_id, co_id, cl_id;
        ClassID    inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].Publi-
cInheritedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ClassNodeNumber;
          if (fr_id !=i){
            output=output+"\nVIOLATION(Predicate5_23): Class("+i+","+j+","+k+") inherits in public from
Class("+fr_id+","+co_id+","+cl_id+") from another framework";
            returnValue=false;
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheri-
tedClasses[ii];
        }
        ii=1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ClassNodeNumber;
          if (fr_id !=i){
            output=output+"\nVIOLATION(Predicate5_23): Class("+i+","+j+","+k+") inherits in protected
from Class("+fr_id+","+co_id+","+cl_id+") from another framework";
            returnValue=false;
          }
```

```
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedIn-
heritedClasses[ii];
        }
        ii=1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ClassNodeNumber;
          if (fr_id !=i){
            output=output+"\nVIOLATION(Predicate5_23): Class("+i+","+j+","+k+") inherits in private from
Class("+fr_id+","+co_id+","+cl_id+") from another framework";
            returnValue=false;
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
        }
      }
    }
  }
  output=output+"\nChecking for predicate5_23 is complete. STATUS: "+returnValue;
  reportViolation(output);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_24(){
  reportViolation("Checking for predicate5_24: SafeInheritance_AmongComponents_WithInTheSameFrame-
work");
  boolean returnValue1=predicate5_30();
  boolean returnValue2=predicate5_31();
  boolean returnValue=returnValue1&&returnValue2;
  reportViolation("Checking for predicate5_24 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_25(){
  reportViolation("Checking for Predicate5_25: SafeInheritance_AmongClasses_WithInTheSameCompon-
ent");
  boolean returnValue1=predicate5_32();
  boolean returnValue2=predicate5_33();
  boolean returnValue=returnValue1&&returnValue2;
  reportViolation("Checking for Predicate5_25 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_26(){
```

```
  reportViolation("Checking for predicate 5_26: SafeInheritance_AmongClasses_WithInTheSharedPools");
  boolean returnValue1=predicate5_34();
  boolean returnValue2=predicate5_35();
  boolean returnValue=returnValue1&&returnValue2;
  reportViolation("Checking for predicate 5_26 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_27(){
  reportViolation("Checking for Predicate 5_27: GoodAssociationOnTheComponentLevel");
  boolean returnValue1=predicate5_36();
  boolean returnValue2=predicate5_37();
  boolean returnValue=returnValue1&&returnValue2;
  reportViolation("Checking for Predicate 5_27 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_28(){
  reportViolation("Checking for Predicate 5_28: GoodAssociationOnInternalClassesLevel");
  boolean returnValue=predicate5_38();
  reportViolation("Checking for Predicate 5_28 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_30(){
  output="Checking                                    for                                    predicate5_30:
A_ComponentMayInheritFromOnlyOneOtherComponentFromTheSameFramework";
  boolean returnValue=true;
  int cnt=0;
  int fr_id, co_id, cl_id;
  for (int i=1; i<=DesignAndImplementationStructure.F;i++){
    for (int j=1; j<=DesignAndImplementationStructure.C[i]; j++){
      for (int k=1; k<=DesignAndImplementationStructure.c[i][j]; k++){
        int ii=1;
        ClassID     inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].Publi-
cInheritedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ClassNodeNumber;
          if (fr_id!=i){
            output=output+"\nVIOLATION(Predicate5_30):     Class("+i+","+j+","+k+")     inherits     from
Class("+fr_id+","+co_id+","+cl_id+") from different framework";
            returnValue=false;
          }
          else{
            if (co_id!=j){
              if     (DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInherited-
Classes[ii].ParentComponent.ComponentNodeNumber !=co_id){
```

```
co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
cnt=cnt+1;
}
}
}
if (cnt>1){
output=output+"\nVIOLATION(Predicate5_30): Component("+i+","+j+") inherits from more than one other components from the same framework";
returnValue=false;
}
ii=ii+1;
inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii];
}
ii=1;
inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii];
while (inhClass!=null){
fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ClassNodeNumber;
if (fr_id!=i){
output=output+"\nVIOLATION(Predicate5_30): Class("+i+","+j+","+k+") inherits from Class("+fr_id+","+co_id+","+cl_id+") from different framework";
returnValue=false;
}
else{
if (co_id!=j){
if (DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ComponentNodeNumber !=co_id){
co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
cnt=cnt+1;
}
}
}
if (cnt>1){
output=output+"\nVIOLATION(Predicate5_30): Component("+i+","+j+") inherits from more than one other components from the same framework";
returnValue=false;
break;
}
ii=ii+1;
inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii];
}
ii=1;
inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[ii];
while (inhClass!=null){
fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
```

```
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ClassNodeNumber;
          if (fr_id!=i){
            output=output+"\nVIOLATION(Predicate5_30):    Class("+i+","+j+","+k+")    inherits    from
Class("+fr_id+","+co_id+","+cl_id+") from different framework";
            returnValue=false;
          }
          else{
            if (co_id!=j){
              if    (DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber !=co_id){
                co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateIn-
heritedClasses[ii].ParentComponent.ComponentNodeNumber;
                cnt=cnt+1;
              }
            }
          }
          if (cnt>1){
            output=output+"\nVIOLATION(Predicate5_30): Component("+i+","+j+") inherits from more than
one other components from the same framework";
            returnValue=false;
            break;
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
        }
      }
    }
  }
  output=output+"\nChecking for predicate5_30 is complete. STATUS: "+returnValue;
  reportViolation(output);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_31(){
  output="Checking                         for                         predicate5_31:
OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework";
  boolean returnValue=true;
  int fr_id, co_id, cl_id;
  for (int i=1; i<=DesignAndImplementationStructure.F;i++){
    for (int j=1; j<=DesignAndImplementationStructure.C[i]; j++){
      for (int k=1; k<=DesignAndImplementationStructure.c[i][j]; k++){
        int ii=1;
        ClassID    inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].Protec-
tedInheritedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ClassNodeNumber;
          if (fr_id!=i){
            output=output+"\nVIOLATION(Predicate5_31):    Class("+i+","+j+","+k+")    inherits    from
Class("+fr_id+","+co_id+","+cl_id+") from different framework";
```

```
            returnValue=false;
          }
          else{
            if (co_id!=j){
              if     (DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheri-
tedClasses[ii].ParentComponent.ComponentNodeNumber !=co_id){
                co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedIn-
heritedClasses[ii].ParentComponent.ComponentNodeNumber;
                output=output+"\nVIOLATION(Predicate5_31: Component("+i+","+j+") inherits in protected
from Component("+i+","+co_id+")";
                returnValue=false;
              }
            }
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedIn-
heritedClasses[ii];
        }
        ii=1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ClassNodeNumber;
          if (fr_id!=i){
            output=output+"\nVIOLATION(Predicate5_31):    Class("+i+","+j+","+k+")    inherits    from
Class("+fr_id+","+co_id+","+cl_id+") from different framework";
            returnValue=false;
          }
          else{
            if (co_id!=j){
              if     (DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber !=co_id){
                co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateIn-
heritedClasses[ii].ParentComponent.ComponentNodeNumber;
                output=output+"\nVIOLATION(Predicate5_31): Component("+i+","+j+") inherits in private
from Component("+i+","+co_id+")";
                returnValue=false;
              }
            }
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
        }
      }
    }
  }
  output=output+"\nChecking for predicate5_31 is complete. STATUS: "+returnValue;
  reportViolation(output);
  return returnValue;
}
/**
 * @return boolean
```

```
 */
protected static boolean predicate5_32(){
  reportViolation("\nChecking for Predicate 5_32: NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent");
  boolean returnValue=true;
  int fr_id, co_id, cl_id;
  for (int i=1; i<=DesignAndImplementationStructure.F;i++){
    for (int j=1; j<=DesignAndImplementationStructure.C[i]; j++){
      int cnt=0;
      for (int k=1; k<=DesignAndImplementationStructure.c[i][j]; k++){
        int ii=1;
        ClassID inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii].ClassNodeNumber;
          if (fr_id!=i){
            reportViolation("\nVIOLATION(Predicate5_32): Class("+i+","+j+","+k+") inherits from Class("+fr_id+","+co_id+","+cl_id+") from different framework");
            cnt=0;
            returnValue=false;
            break;
          }
          else{
            if (co_id==j) cnt=cnt+1;
            if (cnt>1){
              reportViolation("\nVIOLATION(Predicate5_32: Class("+i+","+j+","+k+") has multiple inheritance within the same component");
              cnt=0;
              returnValue=false;
              break;
            }
          }
          ii=ii+1;
          inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PublicInheritedClasses[ii];
        }
        ii=1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii];
        while (inhClass!=null){
          fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ParentFrameworkNodeNumber;
          co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ParentComponent.ComponentNodeNumber;
          cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[ii].ClassNodeNumber;
          if (fr_id!=i){
            reportViolation("\nVIOLATION(Predicate5_32): Class("+i+","+j+","+k+") inherits from Class("+fr_id+","+co_id+","+cl_id+") from different framework");
            cnt=0;
            returnValue=false;
          }
          else{
```

```
          if (co_id==j){
            cnt=cnt+1;
          }
        }
        if (cnt>1){
          reportViolation("\nVIOLATION(Predicate5_32: Class("+i+","+j+","+k+") has multiple inheritance
from other classes within the same component");
          cnt=0;
          returnValue=false;
          break;
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].ProtectedIn-
heritedClasses[ii];
      }
      ii=1;
      inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
      while (inhClass!=null){
        fr_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ParentFrameworkNodeNumber;
        co_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ParentComponent.ComponentNodeNumber;
        cl_id=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInherited-
Classes[ii].ClassNodeNumber;
        if (fr_id!=i){
          reportViolation("\nVIOLATION(Predicate5_32):    Class("+i+","+j+","+k+")    inherits    from
Class("+fr_id+","+co_id+","+cl_id+") from different framework");
          cnt=0;
          returnValue=false;
        }
        else{
          if (co_id==j){
            cnt=cnt+1;
          }
        }
        if (cnt>1){
          reportViolation("\nVIOLATION(Predicate5_32: Class("+i+","+j+","+k+") has multiple inheritance
from other classes within the same component");
          cnt=0;
          returnValue=false;
          break;
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.Frameworks[i].Component[j].Class[k].PrivateInheri-
tedClasses[ii];
      }    }
    }
  }
  reportViolation("\nChecking for predicate5_32 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_33(){
  reportViolation("Checking for Predicate5_33: InheritanceMayBe_Public/Protected/or/Private_WithInThe-
SameComponent");
```

```
  //This predicate is gauranteed by the Structure
  boolean returnValue=true;
  reportViolation("Checking for Predicate5_33 is complete. STATUS: "+returnValue);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_34(){
  output="Checking for predicate 5_34: InheritanceMayNotExistAcrossSharedPools";
  boolean returnValue=true;
  for (int i=1; i<=DesignAndImplementationStructure.S;i++){
    for (int j=1; j<=DesignAndImplementationStructure.SC[i];j++){
      int ii=1;
      SharedClassID    inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInherited-
SharedClasses[ii];
      while (inhClass!=null){
        int          shp_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber;
        int          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii].ClassNodeNumber;
        if (shp_id!=i){
          output=output+"\nVIOLATION(Predicate5_34): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
          returnValue=false;
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii];
      }
      ii=1;
      inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii];
      while (inhClass!=null){
        int        shp_id=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber;
        int        shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii].ClassNodeNumber;
        if (shp_id!=i){
          output=output+"\nVIOLATION(Predicate5_34): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
          returnValue=false;
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii];
      }
      ii=1;
      inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii];
      while (inhClass!=null){
        int          shp_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber;
        int          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii].ClassNodeNumber;
        if (shp_id!=i){
          output=output+"\nVIOLATION(Predicate5_34): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
```

```
          returnValue=false;
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii];
      }
    }
  }
  output=output+"\nChecking for Predicate5_34 is complete. STATUS: "+returnValue;
  reportViolation(output);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_35(){
  output="Checking for predicate 5_35:
NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool";
  boolean returnValue=true;
  for (int i=1; i<=DesignAndImplementationStructure.S;i++){
    for (int j=1; j<=DesignAndImplementationStructure.SC[i];j++){
      int ii=1;
      int cnt=0;
      int shp_id=0;
      int shc_id=0;
      SharedClassID inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInherited-
SharedClasses[ii];
      while (inhClass!=null){
        if (DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedSharedClasses[ii].Pa-
rentSharedPoolNodeNumber!=i){
          shp_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber;
          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii].ClassNodeNumber;
          output=output+"\nVIOLATION(predicate5_35): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
          returnValue=false;
        }
        else{
          cnt=cnt+1;
          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii].ClassNodeNumber;
          if (cnt>1){
            output=output+"\nVIOLATION(Predicate 5_35): SharedClass("+i+","+j+") has multiple inheri-
tance from other shared classes within the same shared pool";
            break;
          }
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PublicInheritedShared-
Classes[ii];
      }
      ii=1;
      cnt=0;
      shc_id=0;
      inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii];
      while (inhClass!=null){
```

```
        if              (DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii].ParentSharedPoolNodeNumber!=i){
          output=output+"\nVIOLATION(predicate5_35): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
          returnValue=false;
        }
        else{
          cnt=cnt+1;
          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii].ClassNodeNumber;
          if (cnt>1){
            output=output+"\nVIOLATION(Predicate 5_35): SharedClass("+i+","+j+") has multiple inheri-
tance from other shared classes within the same shared pool";
            break;
          }
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].ProtectedInheritedShared-
Classes[ii];
      }
      ii=1;
      cnt=0;
      shc_id=0;
      inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii];
      while (inhClass!=null){
        if   (DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedSharedClasses[ii].Pa-
rentSharedPoolNodeNumber!=i){
          output=output+"\nVIOLATION(predicate5_35): SharedClass("+i+","+j+") inherits from Shared-
Class("+shp_id+","+shc_id+") from another shared pool";
          returnValue=false;
        }
        else{
          cnt=cnt+1;
          shc_id=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii].ClassNodeNumber;
          if (cnt>1){
            output=output+"\nVIOLATION(Predicate 5_35): SharedClass("+i+","+j+") has multiple inheri-
tance from other shared classes within the same shared pool";
            break;
          }
        }
        ii=ii+1;
        inhClass=DesignAndImplementationStructure.SharedPools[i].Class[j].PrivateInheritedShared-
Classes[ii];
      }
    }
  }
  output=output+"\nChecking for predicate 5_35 is complete. STATUS: "+returnValue;
  reportViolation(output);
  return returnValue;
}
/**
 * @return boolean
 */
protected static boolean predicate5_36(){
  boolean returnValue=true;
  reportViolation("Checking                    for                    predicate5_36:
```

```
ComponentsMayHaveOnlyUniDirectionalAssociationWithClassesFromTheSharedPools");
    // This requires cross checking the classes of components and classes in the sharedpools. Left for further expan-
sion
    reportViolation("Checking for predicate5_36 is complete. STATUS: "+returnValue);
    return returnValue;
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_37(){
    reportViolation("Checking for predicate5_37: AssociationMayNotExistAcrossFrameworks");
    boolean returnValue=true;
    for (int i=1; i<=DesignAndImplementationStructure.F;i++){
      for (int ii=1; ii<=DesignAndImplementationStructure.F;ii++){
        if (ii==i){ii=ii+1;}
        if (ii>DesignAndImplementationStructure.F){break;}
        if (Associates_FrameworkLevel(i,ii)){
          returnValue=false;
          reportViolation("VIOLATION(Predicate5_37):    Framework("+i+")    associates    with    Frame-
work("+ii+")");}
      }
    }
    reportViolation("Checking for predicate5_37 is complete. STATUS: "+returnValue);
    return returnValue;
  }
  /**
   * @return boolean
   */
  protected static boolean predicate5_38(){
    reportViolation("Checking                                    for                                    predi-
cate5_38:AssociationAndInheritanceMayNotExistBetweenTwoClassesAtTheSameTime");
    boolean returnValue=true;
    for (int i=1; i<=DesignAndImplementationStructure.F;i++){
      for (int j=1; j<=DesignAndImplementationStructure.C[i];j++){
        for (int k=1; k<=DesignAndImplementationStructure.c[i][j];k++){
          for (int ii=1; ii<=DesignAndImplementationStructure.F;ii++){
            if (ii==i){ii=ii+1;}
            if (ii>DesignAndImplementationStructure.F){break;}
            for (int jj=1; jj<=DesignAndImplementationStructure.C[ii];jj++){
              if (jj==j){jj=jj+1;}
              if (jj>DesignAndImplementationStructure.C[ii]){break;}
              for (int kk=1; kk<=DesignAndImplementationStructure.c[ii][jj];kk++){
                if (kk==k){kk=kk+1;}
                if (kk>DesignAndImplementationStructure.c[ii][jj]){break;}
                if (Inherits_ClassLevel(i,j,k,ii,jj,kk) && Associates_ClassLevel(i,j,k,ii,jj,kk) ){
                  returnValue=false;
                  break;
                }
              }
            }
          }
        }
      }
    }
    reportViolation("Checking for predicate5_38 is complete. STATUS: "+returnValue);
    return returnValue;
  }
  private static void reportOutput(String output){
```

```
    System.out.println("\n"+output+"\n");
    return;
  }
  private static void reportViolation(String output){
    System.out.println("\n"+output+"\n");
    return;
  }
}
```

## B1.2.2. Globals.java

Tthis file contains the globals variables used for the implementation of this view's structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: Globals.java
 */
public class Globals {  public static int EXAMPLE_NOM;
  public static int MAX_FR=16;           // maximum number of frameworks -1
  public static int MAX_CO=46;           // maximum number of components -1 per framework
  public static int MAX_CL=25;           // maximum number of classes per component
  public static int MAX_PUB_INH_CL=6;    // maximum number of publicly inherited classes by a one class
  public static int MAX_PRO_INH_CL=6;    // maximum number of protectly inherited classes by a one class;
it should be 0
  public static int MAX_PRV_INH_CL=6;    // maximum number of privately inherited classes by a one class;
it should be 0

  public static int MAX_PUB_OBJ=6;       // maximum number of objects per public section of a class; it should
be 0
  public static int MAX_PUB_MTD=6;       // maximum number of methods per public section of a class
  public static int MAX_PUB_INH_MTD=6; // maximum number of methods inherited by a class;
  public static int MAX_PRO_OBJ=6;       // maximum number of objects per public section of a class
  public static int MAX_PRO_MTD=6;       // maximum number of methods per public section of a class
  public static int MAX_PRO_INH_MTD=6; // maximum number of methods inherited by a class; it should be
0
  public static int MAX_PRV_OBJ=6;       // maximum number of objects per public section of a class
  public static int MAX_PRV_MTD=6;       // maximum number of methods per public section of a class
  public static int MAX_PRV_INH_MTD=6; // maximum number of methods inherited by a class; it should be
0
  public static int MAX_IN_ARGS=4;  // maximum number of input arguments per method
  public static int MAX_OUT_ARGS=4;     // maximum number of output arguments per method

  public static int MAX_SP=10;           // maximum number of shared pools
  public static int MAX_SH_CL=10;      // maximum number of shared classes per shared pool
  public static int MAX_SH_ENT=10;      // maximum number of shared entities per shared pool
  public static int MAX_SH_PUB_INH_CL=6; // maximum number of publicly inherited classes by a one class
  public static int MAX_SH_PRO_INH_CL=6; // maximum number of protectly inherited classes by a one
```

```
class; it should be 0
  public static int MAX_SH_PRV_INH_CL=6; // maximum number of privately inherited classes by a one
class; it should be 0   public static int MAX_SH_PUB_OBJ=6;
  public static int MAX_SH_PUB_MTD=6;
  public static int MAX_SH_PUB_INH_MTD=6;

  public static int MAX_SH_PRO_OBJ=6;
  public static int MAX_SH_PRO_MTD=6;
  public static int MAX_SH_PRO_INH_MTD=6;

  public static int MAX_SH_PRV_OBJ=6;
  public static int MAX_SH_PRV_MTD=6;
  public static int MAX_SH_PRV_INH_MTD=6;
}
```

### B1.2.3. MethodARGS.java

This file contains the code that used to represent a method arguments. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: MethodARGS.java
 */
public class MethodARGS {   String IN_ArgName[] = new String[Globals.MAX_IN_ARGS];
  String IN_ArgType[] = new String[Globals.MAX_IN_ARGS];
  String OUT_ArgName[] = new String[Globals.MAX_OUT_ARGS];
  String OUT_ArgType[] = new String[Globals.MAX_OUT_ARGS];
}
```

### B1.2.4. MethodID.java

This file contains the code used to represent a method ID. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
```

```
 * File: MethodID.java
 */
public class MethodID {
  ClassID ParentClass = new ClassID();
  int MethodNodeNumber;
}
```

### B1.2.5. MethodNode.java

This file contains the code used to represent a method node in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: MethodNode.java
 */
public class MethodNode {
  MethodID ID=new MethodID();
  MethodARGS ARGS=new MethodARGS();
  String ReturnedValueType=null;
  String MethodName=null;public MethodNode() {super();}
}
```

### B1.2.6. PublicSectionNode.java

This file contains the code used to represent a public section node in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: PublicSectionNode.java
 */
public class PublicSectionNode {
  String Object[]=new String[Globals.MAX_PUB_OBJ];
  String ObjectType[]= new String[Globals.MAX_PUB_OBJ];
  MethodNode Method[]=new MethodNode[Globals.MAX_PUB_MTD];public PublicSectionNode() {super();}
}
```

### B1.2.7. ProtectedSectionNode.java

This file contains the code used to represent a protected section node in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: ProtectedSectionNode.java
 */
public class ProtectedSectionNode {
  String Object[]=new String[Globals.MAX_PRO_OBJ];
  String ObjectType[]= new String[Globals.MAX_PRO_OBJ];
  MethodNode Method[]=new MethodNode[Globals.MAX_PRO_MTD];public ProtectedSectionNode() {
    super();
  }
}
```

### B1.2.8. PrivateSectionNode.java

This file contains the code that used to represent a private section node. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: PrivateSectionNode.java
 */
public class PrivateSectionNode {
  String Object[]=new String[Globals.MAX_PRV_OBJ];
  String ObjectType[]= new String[Globals.MAX_PRV_OBJ];
  MethodNode Method[]=new MethodNode[Globals.MAX_PRV_MTD];public PrivateSectionNode() {
  super();
}
}
```

## B1.2.9. ClassID.java

This file contains the code that used to represent a class ID. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: ClassID.java
 */
public class ClassID {
  ComponentID ParentComponent = new ComponentID();
  int ClassNodeNumber;
  String ClassName;
}
```

## B1.2.10. ClassNode.java

This file contains the code that used to represent a class node. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: ClassNode.java
 */
public class ClassNode {   ClassID ID = new ClassID();
  ClassID PublicInheritedClasses[] = new ClassID[Globals.MAX_PUB_INH_CL];
  ClassID ProtectedInheritedClasses[] = new ClassID[Globals.MAX_PRO_INH_CL];
  ClassID PrivateInheritedClasses[] = new ClassID[Globals.MAX_PRV_INH_CL];
  PublicSectionNode PublicSection =new PublicSectionNode();
  ProtectedSectionNode ProtectedSection =new ProtectedSectionNode();
  PrivateSectionNode PrivateSection =new PrivateSectionNode();
  SharedClassID PublicInheritedSharedClasses[] = new SharedClassID[Globals.MAX_SH_PUB_INH_CL];
  SharedClassID          ProtectedInheritedSharedClasses[]          =          new          SharedClassID[Glo-
bals.MAX_SH_PRO_INH_CL];
  SharedClassID          PrivateInheritedSharedClasses[]          =          new          SharedClassID[Glo-
bals.MAX_SH_PRV_INH_CL];public ClassNode() {super();}
}
```

### B1.2.11. ComponentID.java

This file contains the code that used to represent a component ID arguments. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: ComponentID.java
 */
public class ComponentID {
  int ParentFrameworkNodeNumber;
  int ComponentNodeNumber;
}
```

### B1.2.12. ComponentNode.java

This file contains the code used to represent a component node in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: ComponentNode.java
 */
public class ComponentNode {  ComponentID ID = new ComponentID();
  String MainSection=null;
  ClassNode Class[]=new ClassNode[Globals.MAX_CL];public ComponentNode() {super();}
}
```

### B1.2.13. FrameworkNode.java

This file contains the code that used to represent a framework node. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
```

```
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: FrameworkNode.java
 */
public class FrameworkNode {  int FrameworkNodeNumber;
  ComponentNode Component[]=new ComponentNode[Globals.MAX_CO];public FrameworkNode() {
  super();
 }
 }
```

### B1.2.14. SharedMethodID.java

This file contains the code used to represent a a shared method ID. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedMethodID.java
 */
public class SharedMethodID {
  SharedClassID ParentSharedClass = new SharedClassID();
  int MethodNodeNumber;
}
```

### B1.2.15. SharedMethodNode.java

This file contains the code used to represent a a shared method node. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedMethodNode.java
```

```
*/
public class SharedMethodNode {
  SharedMethodID ID=new SharedMethodID();
  MethodARGS ARGS=new MethodARGS();
  String ReturnedValueType=null;
  String MethodName=null;public SharedMethodNode() {super();}
}
```

### B1.2.16. SharedPublicSectionNode.java

This file contains the code used to represent a a shared class public section. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedPublicSectionNode.java
 */
public class SharedPublicSectionNode {
  String Object[]=new String[Globals.MAX_PUB_OBJ];
  String ObjectType[]= new String[Globals.MAX_PUB_OBJ];
  SharedMethodNode Method[]=new SharedMethodNode[Globals.MAX_PUB_MTD];public SharedPu-
blicSectionNode() {super();}
}
```

### B1.2.17. SharedProtectedSectionNode.java

This file contains the code used to represent a a shared class protected section. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto–type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedProtectedSectionNode.java
 */
public class SharedProtectedSectionNode {
  String Object[]=new String[Globals.MAX_SH_PRO_OBJ];
  String ObjectType[]= new String[Globals.MAX_SH_PRO_OBJ];
  SharedMethodNode Method[]=new SharedMethodNode[Globals.MAX_SH_PRO_MTD];public SharedPro-
```

```
tectedSectionNode() {super();}
}
```

### B1.2.18. SharedPrivateSectionNode.java

This file contains the code used to represent a a shared class private section. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedPrivateSectionNode.java
 */
public class SharedPrivateSectionNode {
  String Object[]    =new String[Globals.MAX_SH_PRV_OBJ];
  String ObjectType[]  =   new String[Globals.MAX_SH_PRV_OBJ];
  SharedMethodNode Method[] = new  SharedMethodNode[Globals.MAX_SH_PRV_MTD];public  SharedPri-
vateSectionNode() {super();}
}
```

### B1.2.19. SharedClassID.java

This file contains the code used to represent a shared class ID in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedClassID.java
 */
public class SharedClassID {
  int ParentSharedPoolNodeNumber;   // parent shared pool
  int ClassNodeNumber;
}
```

### B1.2.20. SharedClassNode.java

This file contains the code used to represent a shared class node in the structure. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedClassNode.java
 */
public class SharedClassNode {   SharedClassID ID = new SharedClassID();
  String ClassName;
  ClassID PublicInheritedClasses[] = new ClassID[Globals.MAX_PUB_INH_CL];
  ClassID ProtectedInheritedClasses[] = new ClassID[Globals.MAX_PRO_INH_CL];
  ClassID PrivateInheritedClasses[] = new ClassID[Globals.MAX_PRV_INH_CL];
  SharedClassID PublicInheritedSharedClasses[] = new SharedClassID[Globals.MAX_SH_PUB_INH_CL];
  SharedClassID          ProtectedInheritedSharedClasses[]          =          new          SharedClassID[Glo-
bals.MAX_SH_PRO_INH_CL];
  SharedClassID PrivateInheritedSharedClasses[] = new SharedClassID[Globals.MAX_SH_PRV_INH_CL];
  SharedPublicSectionNode PublicSection =new SharedPublicSectionNode();
  SharedProtectedSectionNode ProtectedSection =new SharedProtectedSectionNode();
  SharedPrivateSectionNode PrivateSection =new SharedPrivateSectionNode();public SharedClassNode() {
    super();
  }
}
```

### B1.2.21. SharedPoolNode.java

This file contains the code that used to represent a shared pool node. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: SharedPoolNode.java
 */
public class SharedPoolNode {  int SharedPoolNodeNumber;
  SharedClassNode Class[]=new SharedClassNode[Globals.MAX_SH_CL];public SharedPoolNode() {
  super();
}
}
```

## B1.2.22. DesignAndImplementationStructure.java

This file contains the implementation of the Design and Implementation Structure. Users should notice that the three examples are imbedded in this class. Next is the source code listing for this file:

```
package V2_DesignAndImplementation_R3;
/**
 * Author: Mohammad Radaideh
 * Software Engineering Research Group
 * Department of Electrical & Computer Engineering
 * McMaster University
 * Hamilton, Ontario
 * January 1999
 * Proto-type Tool View#2 For CO_Toolkits Design&Implementation — Chapter 5 of My Ph.D. Thesis
 * File: DesignAndImplementationStructure.java
 */
public class DesignAndImplementationStructure {
// F: number of frameworks in the system
// C[]: to hold number of components per framework
// c[][]: to hold numbers of classes per component per framework
// className[][][]: to hold names of classes... only ACE example uses this
// MP[][][]: to hold numbers of methods in the public section per class per component per framework
// MR[][][]: to hold numbers of methods in the protected section per class per component per fraemwork
// MV[][][]: to hold numbers of methods in the private section per class per component per framework
// MinhP[][][]: to hold numbers of inherited methods in the public section per class per component per framework
// MinhR[][][]: to hold numbers of inheirted methods in the protected section per class per component per frame-
work
// MinhV[][][]: to hold numbers of inheirted methods in the private section per class per component per frame-
work
// OP[][][]: to hold numbers of objects in the public section per class per component per framework
// OMR[][][]: to hold numbers of objects in the protected section per class per component per fraemwork
// OV[][][]: to hold numbers of objects in the private section per class per component per framework
// OinhP[][][]: to hold numbers of inherited objectss in the public section per class per component per framework
// OinhR[][][]: to hold numbers of inheirted objectss in the protected section per class per component per frame-
work
// OinhV[][][]: to hold numbers of inheirted objectss in the private section per class per component per framework
public static int   F;  // NumberOfFrameworks;
  public static int   C[]     =new int[Globals.MAX_FR];
// to hold number of components per framework
  public static int   c[][]   =new int[Globals.MAX_FR][Globals.MAX_CO];
  public static String  className[][][]= new          String[Globals.MAX_FR][Globals.MAX_CO][Glo-
bals.MAX_CL];
  public static int MP[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MR[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MV[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhP[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhPS[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhR[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhRS[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhV[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int MinhVS[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int OP[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int OR[][][]  =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
```

```
   public static int OV[][][]=new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
//public static int OinhP[][][] =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
//public static int OinhR[][][] =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
//public static int OinhV[][][] =new int[Globals.MAX_FR][Globals.MAX_CO][Globals.MAX_CL];
  public static int   S; // NumberOfSharedPools;
  public static int SC[]     =  new int[Globals.MAX_SP];
  public static int SMP[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SMR[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SMV[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SMinhP[][] =new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SMinhPF[][]  =new int[Globals.MAX_SP][Globals.MAX_CL];
  public static int SMinhR[][] =new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SMinhRF[][]  =new int[Globals.MAX_SP][Globals.MAX_CL];
  public static int SMinhV[][] =new int[Globals.MAX_SP][Globals.MAX_SH_CL];

  public static int SMinhVF[][]  =new int[Globals.MAX_SP][Globals.MAX_CL];

  public static int SOP[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];
  public static int SOR[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];

  public static int SOV[][]   =   new int[Globals.MAX_SP][Globals.MAX_SH_CL];
//public static int SOinhP[][]  =new int[Globals.MAX_SP][Globals.MAX_SH_CL];
//public static int SOinhR[][]  =new int[Globals.MAX_SP][Globals.MAX_SH_CL];
//public static int SOinhV[][]  =new int[Globals.MAX_SP][Globals.MAX_SH_CL];

  private static String output;
  public static FrameworkNode[] Frameworks=new FrameworkNode[Globals.MAX_FR];
  public static SharedPoolNode[] SharedPools = new SharedPoolNode[Globals.MAX_SP];

protected static void display( ) {
  reportStructure("Design and Implementation Structure: As Extracted From The Current Example");
  displayFrameworks();
  displaySharedPools();
  reportStructure("=============================================================================\n");
  return;
}
protected static void displayFrameworks() { output="\n";
  for (int i=1;i<=F;i++){
    reportStructure("\nFramework("+i+")");
    for (int j=1;j<=C[i];j++){
      reportStructure("\n\tComponent("+i+","+j+")");
      for (int k=1;k<=c[i][j];k++){
        String ClassName=Frameworks[i].Component[j].Class[k].ID.ClassName;
        reportStructure("\n\t\tClass("+i+","+j+","+k+"): "+ ClassName);
        // public inheritance
        if (MinhP[i][j][k]>0){
          reportStructure("\n\t\t\tPublic Inheritance From Classes:\t");
          for (int kk=1; kk<=MinhP[i][j][k];kk++){
            int cln= Frameworks[i].Component[j].Class[k].PublicInheritedClasses[kk].ClassNodeNumber;
            int       con=        Frameworks[i].Component[j].Class[k].PublicInheritedClasses[kk].ParentCompo-
nent.ComponentNodeNumber;
            int   frn=  Frameworks[i].Component[j].Class[k].PublicInheritedClasses[kk].ParentComponent.Pa-
rentFrameworkNodeNumber;
            reportStructure("\tClass("+frn+","+con+","+cln+"):"+Frameworks[frn].Compo-
nent[con].Class[cln].ID.ClassName);
          }
        }
        if (MinhPS[i][j][k]>0){
```

```
        for(int kk=1;kk<=MinhPS[i][j][k];kk++){
          int cln= Frameworks[i].Component[j].Class[k].PublicInheritedSharedClasses[kk].ClassNodeNumber;
          int con= Frameworks[i].Component[j].Class[k].PublicInheritedSharedClasses[kk].ParentSharedPoolNodeNumber;
          reportStructure("\tSharedClass("+con+","+cln+"):"+SharedPools[con].Class[cln].ClassName);
        }
      }
      // protected inheritance
      if (MinhR[i][j][k]>0){
        reportStructure("\n\t\t\tProtected Inheritance From Classes:\t");
        for (int kk=1;kk<=MinhR[i][j][k];kk++){
          int cln= Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[kk].ClassNodeNumber;
          int con= Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[kk].ParentComponent.ComponentNodeNumber;
          int frn= Frameworks[i].Component[j].Class[k].ProtectedInheritedClasses[kk].ParentComponent.ParentFrameworkNodeNumber;
          reportStructure("\tClass("+frn+","+con+","+cln+"):"+Frameworks[frn].Component[con].Class[cln].ID.ClassName);
        }
      }
      if(MinhRS[i][j][k]>0){
        for (int kk=1; kk<=MinhRS[i][j][k]; kk++){
          int cln= Frameworks[i].Component[j].Class[k].ProtectedInheritedSharedClasses[kk].ClassNodeNumber;
          int con= Frameworks[i].Component[j].Class[k].ProtectedInheritedSharedClasses[kk].ParentSharedPoolNodeNumber;
          reportStructure("\tSharedClass("+con+","+cln+"):"+SharedPools[con].Class[cln].ClassName);
        }
      }
      // private inheritance
      if (MinhV[i][j][k]>0){
        reportStructure("\n\t\t\tPrivate Inheritance From Classes:\t");
        for (int kk=1; kk<=MinhV[i][j][k];kk++){
          int cln= Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[kk].ClassNodeNumber;
          int con= Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[kk].ParentComponent.ComponentNodeNumber;
          int frn= Frameworks[i].Component[j].Class[k].PrivateInheritedClasses[kk].ParentComponent.ParentFrameworkNodeNumber;
          reportStructure("\tClass("+frn+","+con+","+cln+"):"+Frameworks[frn].Component[con].Class[cln].ID.ClassName);
        }
      }
      if(MinhVS[i][j][k]>0){
        for (int kk=1; kk<=MinhVS[i][j][k]; kk++){
          int cln= Frameworks[i].Component[j].Class[k].PrivateInheritedSharedClasses[kk].ClassNodeNumber;
          int con= Frameworks[i].Component[j].Class[k].PrivateInheritedSharedClasses[kk].ParentSharedPoolNodeNumber;
          reportStructure("\tSharedClass("+con+","+cln+"):"+SharedPools[con].Class[cln].ClassName);
        }
      }
      //public section
      reportStructure("\n\t\t\tPublic Section Methods & Objects");
      for (int l=1; l<=MP[i][j][k]; l++){
        String MethodName=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].MethodName;
        reportStructure("\n\t\t\t\tMethodName: "+MethodName);
        //ReturnedValueType
```

```
        String    ReturnedValueType=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].Retur-
nedValueType;
        reportStructure("\n\t\t\t\t\tReturnedValueType:\t\t"+ReturnedValueType);
        //IN_ARGS
        reportStructure("\n\t\t\t\t\tIN_ArgNames(Types): ");
        int ll=1;
        String    IN_ARG=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ARGS.IN_Arg-
Name[ll];
        String                IN_ARGType=Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ARGS.IN_ArgType[ll];
        while (IN_ARG!=null){
          reportStructure("\t"+IN_ARG+"("+IN_ARGType+")");
          ll=ll+1;
          IN_ARG=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ARGS.IN_Arg-
Name[ll];
          IN_ARGType=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ARGS.IN_Arg-
Type[ll];
        }
        //OUT_ARGS
        reportStructure("\n\t\t\t\t\tOUT_ArgNames(Types): ");
        ll=1;
        String                OUT_ARG=Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ARGS.OUT_ArgName[ll];
        String            OUT_ARGType=Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        while (OUT_ARG!=null){
          reportStructure("\t"+OUT_ARG+"("+OUT_ARGType+")");
          ll=ll+1;
          OUT_ARG=Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ARGS.OUT_Arg-
Name[ll];
          OUT_ARGType=Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        }
      }
      //objects of in public section
      output=output+"\n\t\t\t\tObjects: ";
      int ll=1;
      String ObjectName=Frameworks[i].Component[j].Class[k].PublicSection.Object[ll];
      String ObjectType=Frameworks[i].Component[j].Class[k].PublicSection.ObjectType[ll];
      while (ObjectName!=null){
        output=output+"\t"+ObjectName+"("+ObjectType+")";
        ll=ll+1;
        ObjectName=Frameworks[i].Component[j].Class[k].PublicSection.Object[ll];
        ObjectType=Frameworks[i].Component[j].Class[k].PublicSection.ObjectType[ll];
      }
      //protected section
      reportStructure("\n\t\t\tProtected Section Methods & Objects");
      for (int l=1; l<=MR[i][j][k]; l++){
        String    MethodName=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].Method-
Name;
        reportStructure("\n\t\t\t\tMethodName: "+MethodName);
        //ReturnedValueType
        String    ReturnedValueType=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].Re-
turnedValueType;
        reportStructure("\n\t\t\t\t\tReturnedValueType:\t\t"+ReturnedValueType);
        //IN_ARGS
        reportStructure("\n\t\t\t\t\tIN_ArgNames(Types): ");
        ll=1;
```

```
        String   IN_ARG=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ARGS.IN_Arg-
Name[ll];
        String                    IN_ARGType=Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ARGS.IN_ArgType[ll];
        while (IN_ARG!=null){
          reportStructure("\t"+IN_ARG+"("+IN_ARGType+")");
          ll=ll+1;
          IN_ARG=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ARGS.IN_Arg-
Name[ll];
          IN_ARGType=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ARGS.IN_Arg-
Type[ll];
        }
        //OUT_ARGS
        reportStructure("\n\t\t\t\t\tOUT_ArgNames(Types): ");
        ll=1;
        String                   OUT_ARG=Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ARGS.OUT_ArgName[ll];
        String               OUT_ARGType=Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        while (OUT_ARG!=null){
          reportStructure("\t"+OUT_ARG+"("+OUT_ARGType+")");
          ll=ll+1;
          OUT_ARG=Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ARGS.OUT_Arg-
Name[ll];
          OUT_ARGType=Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        }
      }
      //objects of in protected section
      reportStructure("\n\t\t\t\tObjects: ");
      ll=1;
      ObjectName=Frameworks[i].Component[j].Class[k].ProtectedSection.Object[ll];
      ObjectType=Frameworks[i].Component[j].Class[k].ProtectedSection.ObjectType[ll];
      while (ObjectName!=null){
        reportStructure("\t"+ObjectName+"("+ObjectType+")");
        ll=ll+1;
        ObjectName=Frameworks[i].Component[j].Class[k].ProtectedSection.Object[ll];
        ObjectType=Frameworks[i].Component[j].Class[k].ProtectedSection.ObjectType[ll];
      }
      //private section
      reportStructure("\n\t\t\tPrivate Section Methods & Objects");
      for (int l=1; l<=MV[i][j][k]; l++){
        String MethodName=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].MethodName;
        reportStructure("\n\t\t\t\tMethodName: "+MethodName);
        //ReturnedValueType
        String    ReturnedValueType=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].Retur-
nedValueType;
        reportStructure("\n\t\t\t\t\tReturnedValueType:\t\t"+ReturnedValueType);
        //IN_ARGS
        reportStructure("\n\t\t\t\t\tIN_ArgNames(Types): ");
        ll=1;
        String     IN_ARG=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ARGS.IN_Arg-
Name[ll];
        String                    IN_ARGType=Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ARGS.IN_ArgType[ll];
        while (IN_ARG!=null){
          reportStructure("\t"+IN_ARG+"("+IN_ARGType+")");
          ll=ll+1;
```

```
          IN_ARG=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ARGS.IN_Arg-
Name[ll];
          IN_ARGType=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ARGS.IN_Arg-
Type[ll];
        }
        //OUT_ARGS
        reportStructure("\n\t\t\t\t\tOUT_ArgNames(Types): ");
        ll=1;
        String                          OUT_ARG=Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ARGS.OUT_ArgName[ll];
        String                      OUT_ARGType=Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        while (OUT_ARG!=null){
          reportStructure("\t"+OUT_ARG+"("+OUT_ARGType+")");
          ll=ll+1;
          OUT_ARG=Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ARGS.OUT_Arg-
Name[ll];
          OUT_ARGType=Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ARGS.OUT_ArgType[ll];
        }
      }
      //objects of in private section
      reportStructure("\n\t\t\t\tObjects: ");
      ll=1;
      ObjectName=Frameworks[i].Component[j].Class[k].PrivateSection.Object[ll];
      ObjectType=Frameworks[i].Component[j].Class[k].PrivateSection.ObjectType[ll];
      while (ObjectName!=null){
        reportStructure("\t"+ObjectName+"("+ObjectType+")");
        ll=ll+1;
        ObjectName=Frameworks[i].Component[j].Class[k].PrivateSection.Object[ll];
        ObjectType=Frameworks[i].Component[j].Class[k].PrivateSection.ObjectType[ll];
      }
    }
  }
 }
 reportStructure("\n\n");
}
protected static void displaySharedPools() { output="\n";
 for (int i=1;i<=S;i++){
   output=output+"\n\nSharedPool("+i+")\n\n";
   for (int j=1;j<=SC[i];j++){
     String ClassName=SharedPools[i].Class[j].ClassName;
     output=output+"\tClass("+i+","+j+"): " + ClassName+"\n\n";
     // public inheritance
     if (SMinhP[i][j]>0){
       output=output+"\t\tPublic Inheritance From Classes:\t";
       for (int k=1; k<=SMinhP[i][j];k++){
         int cln= SharedPools[i].Class[j].PublicInheritedSharedClasses[k].ClassNodeNumber;
         int con= SharedPools[i].Class[j].PublicInheritedSharedClasses[k].ParentSharedPoolNodeNumber;
         output=output+"\tSharedClass("+con+","+cln+")";
       }
     }
     output=output+"\n\n";
     if (SMinhPF[i][j]>0){
       for(int k=1;k<=SMinhPF[i][j];k++){
         int cln= SharedPools[i].Class[j].PublicInheritedClasses[k].ClassNodeNumber;
         int  con=  SharedPools[i].Class[j].PublicInheritedClasses[k].ParentComponent.ComponentNodeNum-
ber;
```

```
        int frn= SharedPools[i].Class[j].PublicInheritedClasses[k].ParentComponent.ParentFrameworkNode-
Number;
        output=output+"\tClass("+frn+","+con+","+cln+")";
      }
    }
    // protected inheritance
    if (SMinhR[i][j]>0){
      output=output+"\t\tProtected Inheritance From Classes:\t";
      for (int k=1;k<=SMinhR[i][j];k++){
        int cln= SharedPools[i].Class[j].ProtectedInheritedSharedClasses[k].ClassNodeNumber;
        int con= SharedPools[i].Class[j].ProtectedInheritedSharedClasses[k].ParentSharedPoolNodeNumber;
        output=output+"\tSharedClass("+con+","+cln+")";
      }
    }
    if(SMinhRF[i][j]>0){
      for (int k=1; k<=SMinhRF[i][j]; k++){
        int cln= SharedPools[i].Class[j].ProtectedInheritedClasses[k].ClassNodeNumber;
        int con= SharedPools[i].Class[j].ProtectedInheritedClasses[k].ParentComponent.ComponentNode-
Number;
        int frn= SharedPools[i].Class[j].ProtectedInheritedClasses[k].ParentComponent.ParentFramework-
NodeNumber;
        output=output+"\tClass("+frn+","+con+","+cln+")";
      }
    }
    output=output+"\n\n";
    // private inheritance
    if (SMinhV[i][j]>0){
      output=output+"\t\tPrivate Inheritance From Classes:\t";
      for (int k=1; k<=SMinhV[i][j];k++){
        int cln= SharedPools[i].Class[j].PrivateInheritedSharedClasses[k].ClassNodeNumber;
        int con= SharedPools[i].Class[j].PrivateInheritedSharedClasses[k].ParentSharedPoolNodeNumber;
        output=output+"\tSharedClass("+con+","+cln+")";
      }
    }

    if(SMinhVF[i][j]>0){
      for (int k=1; k<=SMinhVF[i][j]; k++){
        int cln= SharedPools[i].Class[j].PrivateInheritedClasses[k].ClassNodeNumber;
        int con= SharedPools[i].Class[j].PrivateInheritedClasses[k].ParentComponent.ComponentNodeNum-
ber;
        int frn= SharedPools[i].Class[j].PrivateInheritedClasses[k].ParentComponent.ParentFrameworkNo-
deNumber;
        output=output+"\tClass("+frn+","+con+","+cln+")";
      }
    }
    output=output+"\n\n";

    //public section
    output=output+"\t\tPublic Section Methods & Objects\n";
    for (int l=1; l<=SMP[i][j]; l++){
      String MethodName=SharedPools[i].Class[j].PublicSection.Method[l].MethodName;
      output=output+"\t\t\tMethodName: "+MethodName+"\n";
      //ReturnedValueType
      String ReturnedValueType=SharedPools[i].Class[j].PublicSection.Method[l].ReturnedValueType;
      output=output+"\t\t\t\tReturnedValueType:\t\t"+ReturnedValueType+"\n";
      //IN_ARGS
      output=output+"\t\t\t\tIN_ArgNames(Types): ";
      int ll=1;
```

```
    String IN_ARG=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.IN_ArgName[ll];
    String IN_ARGType=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.IN_ArgType[ll];
    while (IN_ARG!=null){
      output=output+"\t"+IN_ARG+"("+IN_ARGType+")";
      ll=ll+1;
      IN_ARG=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.IN_ArgName[ll];
      IN_ARGType=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.IN_ArgType[ll];
    }
    //OUT_ARGS
    output=output+"\n\t\t\t\tOUT_ArgNames(Types): ";
    ll=1;
    String OUT_ARG=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.OUT_ArgName[ll];
    String OUT_ARGType=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.OUT_ArgType[ll];
    while (OUT_ARG!=null){
      output=output+"\t"+OUT_ARG+"("+OUT_ARGType+")";
      ll=ll+1;
      OUT_ARG=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.OUT_ArgName[ll];
      OUT_ARGType=SharedPools[i].Class[j].PublicSection.Method[l].ARGS.OUT_ArgType[ll];
    }
    output=output+"\n";
  }
  //objects of in public section
  output=output+"\n\t\t\tObjects: ";
  int ll=1;
  String ObjectName=SharedPools[i].Class[j].PublicSection.Object[ll];
  String ObjectType=SharedPools[i].Class[j].PublicSection.ObjectType[ll];
  while (ObjectName!=null){
    output=output+"\t"+ObjectName+"("+ObjectType+")";
    ll=ll+1;
    ObjectName=SharedPools[i].Class[j].PublicSection.Object[ll];
    ObjectType=SharedPools[i].Class[j].PublicSection.ObjectType[ll];
  }
  output=output+"\n\n";

  //protected section
  output=output+"\t\tProtected Section Methods & Objects\n";
  for (int l=1; l<=SMR[i][j]; l++){
    String MethodName=SharedPools[i].Class[j].ProtectedSection.Method[l].MethodName;
    output=output+"\t\t\tMethodName: "+MethodName+"\n";
    //ReturnedValueType
    String ReturnedValueType=SharedPools[i].Class[j].ProtectedSection.Method[l].ReturnedValueType;
    output=output+"\t\t\t\tReturnedValueType:\t\t"+ReturnedValueType+"\n";
    //IN_ARGS
    output=output+"\t\t\t\tIN_ArgNames(Types): ";
    ll=1;
    String IN_ARG=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.IN_ArgName[ll];
    String IN_ARGType=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.IN_ArgType[ll];
    while (IN_ARG!=null){
      output=output+"\t"+IN_ARG+"("+IN_ARGType+")";
      ll=ll+1;
      IN_ARG=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.IN_ArgName[ll];
      IN_ARGType=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.IN_ArgType[ll];
    }
    //OUT_ARGS
    output=output+"\n\t\t\t\tOUT_ArgNames(Types): ";
    ll=1;
    String OUT_ARG=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.OUT_ArgName[ll];
    String OUT_ARGType=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.OUT_ArgType[ll];
```

```
.
    while (OUT_ARG!=null){
      output=output+"\t"+OUT_ARG+"("+OUT_ARGType+")";
      ll=ll+1;
      OUT_ARG=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.OUT_ArgName[ll];
      OUT_ARGType=SharedPools[i].Class[j].ProtectedSection.Method[l].ARGS.OUT_ArgType[ll];
    }
    output=output+"\n";
  }
  //objects of in protected section
  output=output+"\n\t\t\tObjects: ";
  ll=1;
  ObjectName=SharedPools[i].Class[j].ProtectedSection.Object[ll];
  ObjectType=SharedPools[i].Class[j].ProtectedSection.ObjectType[ll];
  while (ObjectName!=null){
    output=output+"\t"+ObjectName+"("+ObjectType+")";
    ll=ll+1;
    ObjectName=SharedPools[i].Class[j].ProtectedSection.Object[ll];
    ObjectType=SharedPools[i].Class[j].ProtectedSection.ObjectType[ll];
  }
  output=output+"\n\n";

  //private section
  output=output+"\t\tPrivate Section Methods & Objects\n";
  for (int l=1; l<=SMV[i][j]; l++){
    String MethodName=SharedPools[i].Class[j].PrivateSection.Method[l].MethodName;
    output=output+"\t\t\tMethodName: "+MethodName+"\n";
    //ReturnedValueType
    String ReturnedValueType=SharedPools[i].Class[j].PrivateSection.Method[l].ReturnedValueType;
    output=output+"\t\t\tReturnedValueType:\t\t"+ReturnedValueType+"\n";
    //IN_ARGS
    output=output+"\t\t\tIN_ArgNames(Types): ";
    ll=1;
    String IN_ARG=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.IN_ArgName[ll];
    String IN_ARGType=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.IN_ArgType[ll];
    while (IN_ARG!=null){
      output=output+"\t"+IN_ARG+"("+IN_ARGType+")";
      ll=ll+1;
      IN_ARG=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.IN_ArgName[ll];
      IN_ARGType=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.IN_ArgType[ll];
    }
    //OUT_ARGS
    output=output+"\n\t\t\tOUT_ArgNames(Types): ";
    ll=1;
    String OUT_ARG=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.OUT_ArgName[ll];
    String OUT_ARGType=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.OUT_ArgType[ll];
    while (OUT_ARG!=null){
      output=output+"\t"+OUT_ARG+"("+OUT_ARGType+")";
      ll=ll+1;
      OUT_ARG=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.OUT_ArgName[ll];
      OUT_ARGType=SharedPools[i].Class[j].PrivateSection.Method[l].ARGS.OUT_ArgType[ll];
    }
    output=output+"\n\n";
  }
  //objects of in private section
  output=output+"\n\t\t\tObjects: ";
  ll=1;
  ObjectName=SharedPools[i].Class[j].PrivateSection.Object[ll];
  ObjectType=SharedPools[i].Class[j].PrivateSection.ObjectType[ll];
```

```
        while (ObjectName!=null){
          output=output+"\t"+ObjectName+"("+ObjectType+")";
          ll=ll+1;
          ObjectName=SharedPools[i].Class[j].PrivateSection.Object[ll];
          ObjectType=SharedPools[i].Class[j].PrivateSection.ObjectType[ll];
        }
        output=output+"\n\n";
      }
    }
    reportStructure(output);
  }
  protected static void initializeFor_Example_1 ( ) {
    initializeFrameworks_Example_1();
    initializeSharedPools_Example_1();
    display();
    return;
  }
  protected static void initializeFor_Example_2 ( ) {
    initializeFrameworks_Example_2();
    initializeSharedPools_Example_2();
    display();
    return;
  }
  protected static void initializeFor_Example_3_ACE ( ) {
    initializeFrameworks_Example_3_ACE();
    initializeSharedPools_Example_3_ACE();
    display();
    return;
  }
  protected static void initializeFrameworks_Example_1() {
  // ****************************************
  // STEP#1 read values for:
  //       F, C[], c[][],
  //       MP[][][], MR[][][], MV[][][]
  //       OP[][][], OR[][][], OV[][][]
  // ****************************************
  //number of frameworks
    F=2;
  //number of components per framework
    C[1]=2; C[2]=2;
  //number of classes per component per framework
    c[1][1]=2; c[1][2]=2;
    c[2][1]=2; c[2][2]=2;
  //number of public methods per class per component per framework
    MP[1][1][1]=2;  MP[1][1][2]=2;
    MP[1][2][1]=2;  MP[1][2][2]=2;
    MP[2][1][1]=2;  MP[2][1][2]=2;
    MP[2][2][1]=2;  MP[2][2][2]=2;
  //number of protected methods per class per component per framework
    MR[1][1][1]=1;  MR[1][1][2]=1;
    MR[1][2][1]=1;  MR[1][2][2]=1;
    MR[2][1][1]=1;  MR[2][1][2]=1;
    MR[2][2][1]=1;  MR[2][2][2]=1;
  //number of private methods per class per component per framework
    MV[1][1][1]=1;   MV[1][1][2]=1;
    MV[1][2][1]=1;   MV[1][2][2]=1;
    MV[2][1][1]=1;   MV[2][1][2]=1;
```

```
  MV[2][2][1]=1;    MV[2][2][2]=1;
 //number of public objects per class per component per framework
   OP[1][1][1]=0;  OP[1][1][2]=1;
   OP[1][2][1]=1;  OP[1][2][2]=0;
   OP[2][1][1]=1;  OP[2][1][2]=0;
   OP[2][2][1]=0;  OP[2][2][2]=0;
 //number of protected methods per class per component per framework
   OR[1][1][1]=2;  OR[1][1][2]=2;
   OR[1][2][1]=2;  OR[1][2][2]=2;
   OR[2][1][1]=2;  OR[2][1][2]=2;
   OR[2][2][1]=2;  OR[2][2][2]=2;
 //number of private methods per class per component per framework
   OV[1][1][1]=4;  OV[1][1][2]=4;
   OV[1][2][1]=4;  OV[1][2][2]=4;
   OV[2][1][1]=4;  OV[2][1][2]=4;
   OV[2][2][1]=4;  OV[2][2][2]=4;
 //number of classes inherited in public per class per component per framework
   MinhP[1][1][1]=1; MinhP[1][1][2]=0;
   MinhP[1][2][1]=1; MinhP[1][2][2]=0;
   MinhP[2][1][1]=1; MinhP[2][1][2]=0;
   MinhP[2][2][1]=1; MinhP[2][2][2]=0;
 //number of classes inherited in protected per class per component per framework
   MinhR[1][1][1]=0; MinhR[1][1][2]=0;
   MinhR[1][2][1]=0; MinhR[1][2][2]=0;
   MinhR[2][1][1]=0; MinhR[2][1][2]=0;
   MinhR[2][2][1]=0; MinhR[2][2][2]=0;
 //number of classes inherited in private per class per component per framework
   MinhV[1][1][1]=0; MinhV[1][1][2]=0;
   MinhV[1][2][1]=0; MinhV[1][2][2]=0;
   MinhV[2][1][1]=0; MinhV[2][1][2]=0;
   MinhV[2][2][1]=0; MinhV[2][2][2]=0;
 //number of classes inherited from shared pools in public per class per component per framework
   MinhPS[1][1][1]=0;  MinhPS[1][1][2]=0;
   MinhPS[1][2][1]=0;  MinhPS[1][2][2]=0;
   MinhPS[2][1][1]=0;  MinhPS[2][1][2]=0;
   MinhPS[2][2][1]=0;  MinhPS[2][2][2]=0;
 //number of classes inherited from shared pools in protected per class per component per framework
   MinhRS[1][1][1]=0; MinhRS[1][1][2]=0;
   MinhRS[1][2][1]=0; MinhRS[1][2][2]=0;
   MinhRS[2][1][1]=0; MinhRS[2][1][2]=0;
   MinhRS[2][2][1]=0; MinhRS[2][2][2]=0;
 //number of classes inherited  from shared pools in private per class per component per framework
   MinhVS[1][1][1]=0; MinhVS[1][1][2]=0;
   MinhVS[1][2][1]=0; MinhVS[1][2][2]=0;
   MinhVS[2][1][1]=0; MinhVS[2][1][2]=0;
   MinhVS[2][2][1]=0; MinhVS[2][2][2]=0; //
*******************************************************
// STEP#2: instantiate for the Frameworks objects and the embedied Component, MainSection,
// Class, PublicSection, ProtectedSection, PrivateSection, Method, etc. etc, objects
// ALSO: Assign ID values for each object
// *******************************************************
// instantiate the framework objects
  for (int i=1; i<=F; i++){
    Frameworks[i]=new FrameworkNode();
    Frameworks[i].FrameworkNodeNumber=i;

    //instantiate the component objects for this new framework object
    for (int j=1; j<=C[i]; j++){
```

```
        Frameworks[i].Component[j]=new ComponentNode();
        Frameworks[i].Component[j].ID.ComponentNodeNumber=j;
        Frameworks[i].Component[j].ID.ParentFrameworkNodeNumber=i;
        Frameworks[i].Component[j].MainSection="main_"+i+"_"+j;
        //instantiate the class objects for this component object
        for (int k=1; k<=c[i][j]; k++){
          Frameworks[i].Component[j].Class[k]=new ClassNode();
          Frameworks[i].Component[j].Class[k].ID.ClassNodeNumber=k;
          Frameworks[i].Component[j].Class[k].ID.ClassName="class"+i+j+k;
          Frameworks[i].Component[j].Class[k].ID.ParentComponent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].ID.ParentComponent.ParentFrameworkNodeNumber=i;
          //instantiate for the public section of this class
          Frameworks[i].Component[j].Class[k].PublicSection=new PublicSectionNode();
          for (int l=1; l<=MP[i][j][k];l++){
            Frameworks[i].Component[j].Class[k].PublicSection.Method[l]=new MethodNode();
            Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ReturnedValueType="void";
            Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.MethodNodeNumber=l;
            Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
            Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
            Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
          }
          //instantiate for the protected section of this class
          Frameworks[i].Component[j].Class[k].ProtectedSection=new ProtectedSectionNode();
          for (int l=1; l<=MR[i][j][k];l++){
            Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l]=new MethodNode();
            Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ReturnedValueType="void";
            Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.MethodNodeNumber=l;
            Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
            Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
            Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
          }
          //instantiate for the private section of this class
          Frameworks[i].Component[j].Class[k].PrivateSection=new PrivateSectionNode();
          for (int l=1; l<=MV[i][j][k];l++){
            Frameworks[i].Component[j].Class[k].PrivateSection.Method[l]=new MethodNode();
            Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ReturnedValueType="void";
            Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.MethodNodeNumber=l;
            Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
            Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
            Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
          }
        }
      }
    }
// ******************************************************
// STEP#3:  Read the entries for each of these objects
// ******************************************************
//CLASS(1,1,1)
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].MethodName="p_mthd1111";
```

```
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Y";
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="T2";
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgName[1]="Z";
Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgType[1]="T3";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].MethodName="p_mthd1112";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T3";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ReturnedValueType="T4";
Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].MethodName="r_mthd1111";
Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T2";
Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgName[1]="Y";
Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgType[1]="T3";
Frameworks[1].Component[1].Class[1].ProtectedSection.Object[1]="r_obj1112";
Frameworks[1].Component[1].Class[1].ProtectedSection.ObjectType[1]="class112";
Frameworks[1].Component[1].Class[1].ProtectedSection.Object[2]="r_obj1112";
Frameworks[1].Component[1].Class[1].ProtectedSection.ObjectType[2]="T12";

Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].MethodName="v_mthd1111";
Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgName[1]="Z";
Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T1";
Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgName[1]="W";
Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgType[1]="T5";
Frameworks[1].Component[1].Class[1].PrivateSection.Object[1]="v_obj1111";
Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[1]="class221";
Frameworks[1].Component[1].Class[1].PrivateSection.Object[2]="v_obj1112";
Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[2]="class222";
Frameworks[1].Component[1].Class[1].PrivateSection.Object[3]="v_objt1113";
Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[3]="T9";
Frameworks[1].Component[1].Class[1].PrivateSection.Object[4]="v_obj1114";
Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[4]="T5";
//CLASS(1,1,2)
Frameworks[1].Component[1].Class[2].PublicSection.Method[1].MethodName="p_mthd1121";
Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="A";
Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ReturnedValueType="T4";

Frameworks[1].Component[1].Class[2].PublicSection.Method[2].MethodName="p_mthd1122";
Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T3";
Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ReturnedValueType="T4";
Frameworks[1].Component[1].Class[2].PublicSection.Object[1]="p_obj1121";
Frameworks[1].Component[1].Class[2].PublicSection.ObjectType[1]="T1";
Frameworks[1].Component[1].Class[2].ProtectedSection.Method[1].MethodName="r_mthd1121";
Frameworks[1].Component[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
Frameworks[1].Component[1].Class[2].ProtectedSection.Object[1]="r_obj1121";
Frameworks[1].Component[1].Class[2].ProtectedSection.ObjectType[1]="class212";
Frameworks[1].Component[1].Class[2].ProtectedSection.Object[2]="r_obj1122";
Frameworks[1].Component[1].Class[2].ProtectedSection.ObjectType[2]="T5";
Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].MethodName="v_mthd1121";
Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ARGS.IN_ArgName[1]="A";
Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T3";
```

```
Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T3";
Frameworks[1].Component[1].Class[2].PrivateSection.Object[1]="v_obj1121";
Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[1]="class111";
Frameworks[1].Component[1].Class[2].PrivateSection.Object[2]="v_obj1122";
Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[2]="T6";
Frameworks[1].Component[1].Class[2].PrivateSection.Object[3]="v_obj1123";
Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[3]="T7";
Frameworks[1].Component[1].Class[2].PrivateSection.Object[4]="v_obj1124";
Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[4]="T1";
//CLASS(1,2,1)
Frameworks[1].Component[2].Class[1].PublicSection.Method[1].MethodName="p_mthd1211";
Frameworks[1].Component[2].Class[1].PublicSection.Method[2].MethodName="p_mthd1212";
Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="X";
Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.OUT_ArgName[1]="Z";
Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.OUT_ArgType[1]="T3";
Frameworks[1].Component[2].Class[1].PublicSection.Object[1]="pub_object1211";
Frameworks[1].Component[2].Class[1].PublicSection.ObjectType[1]="T11";
Frameworks[1].Component[2].Class[1].ProtectedSection.Method[1].MethodName="r_mthd1211";
Frameworks[1].Component[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T3";
Frameworks[1].Component[2].Class[1].ProtectedSection.Object[1]="r_obj1211";
Frameworks[1].Component[2].Class[1].ProtectedSection.ObjectType[1]="class212";
Frameworks[1].Component[2].Class[1].ProtectedSection.Object[2]="r_obj1212";
Frameworks[1].Component[2].Class[1].ProtectedSection.ObjectType[2]="T10";
Frameworks[1].Component[2].Class[1].PrivateSection.Method[1].MethodName="v_mthd1211";
Frameworks[1].Component[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
Frameworks[1].Component[2].Class[1].PrivateSection.Object[1]="v_obj1211";
Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[1]="class212";
Frameworks[1].Component[2].Class[1].PrivateSection.Object[2]="v_obj1212";
Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[2]="T6";
Frameworks[1].Component[2].Class[1].PrivateSection.Object[3]="v_obj1213";
Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[3]="T2";
Frameworks[1].Component[2].Class[1].PrivateSection.Object[4]="v_obj1214";
Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[4]="class222";
//CLASS(1,2,2)
Frameworks[1].Component[2].Class[2].PublicSection.Method[1].MethodName="p_mthd1221";
Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="Y";
Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T2";
Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ReturnedValueType="T2";
Frameworks[1].Component[2].Class[2].PublicSection.Method[2].MethodName="p_mthd1222";
Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="Y";
Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ReturnedValueType="T1";
Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].MethodName="r_mthd1221";
Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T4";
Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T1";
Frameworks[1].Component[2].Class[2].ProtectedSection.Object[1]="r_obj1221";
Frameworks[1].Component[2].Class[2].ProtectedSection.ObjectType[1]="T11";
Frameworks[1].Component[2].Class[2].ProtectedSection.Object[2]="r_obj1222";
Frameworks[1].Component[2].Class[2].ProtectedSection.ObjectType[2]="T6";
Frameworks[1].Component[2].Class[2].PrivateSection.Method[1].MethodName="v_mthd1221";
Frameworks[1].Component[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T1";
Frameworks[1].Component[2].Class[2].PrivateSection.Object[1]="v_obj1221";
Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[1]="class111";
Frameworks[1].Component[2].Class[2].PrivateSection.Object[2]="v_obj1222";
Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[2]="class221";
Frameworks[1].Component[2].Class[2].PrivateSection.Object[3]="v_obj1223";
```

```
 Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[3]="T12";
 Frameworks[1].Component[2].Class[2].PrivateSection.Object[4]="v_obj1224";
 Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[4]="T8";

//CLASS(2,1,1)
 Frameworks[2].Component[1].Class[1].PublicSection.Method[1].MethodName="p_mthd2111";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T6";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ReturnedValueType="T3";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[2].MethodName="p_mthd2112";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="W";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
 Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ReturnedValueType="T2";
 Frameworks[2].Component[1].Class[1].PublicSection.Object[1]="p_obj2111";
 Frameworks[2].Component[1].Class[1].PublicSection.ObjectType[1]="T12";
 Frameworks[2].Component[1].Class[1].ProtectedSection.Method[1].MethodName="r_mthd2111";
 Frameworks[2].Component[1].Class[1].ProtectedSection.Method[1].ReturnedValueType="T1";
 Frameworks[2].Component[1].Class[1].ProtectedSection.Object[1]="r_obj2111";
 Frameworks[2].Component[1].Class[1].ProtectedSection.ObjectType[1]="class112";
 Frameworks[2].Component[1].Class[1].ProtectedSection.Object[2]="r_obj2112";
 Frameworks[2].Component[1].Class[1].ProtectedSection.ObjectType[2]="T13";
 Frameworks[2].Component[1].Class[1].PrivateSection.Method[1].MethodName="v_mthd2111";
 Frameworks[2].Component[1].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
 Frameworks[2].Component[1].Class[1].PrivateSection.Object[1]="v_obj2111";
 Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[1]="class121";
 Frameworks[2].Component[1].Class[1].PrivateSection.Object[2]="v_obj2112";
 Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[2]="class122";
 Frameworks[2].Component[1].Class[1].PrivateSection.Object[3]="v_obj2113";
 Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[3]="T7";
 Frameworks[2].Component[1].Class[1].PrivateSection.Object[4]="v_obj2114";
 Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[4]="T6";

//CLASS(2,1,2)
 Frameworks[2].Component[1].Class[2].PublicSection.Method[1].MethodName="p_mthd2121";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ReturnedValueType="T1";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[2].MethodName="p_mthd2122";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="W";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T2";
 Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ReturnedValueType="T3";
 Frameworks[2].Component[1].Class[2].ProtectedSection.Method[1].MethodName="r_mthd2121";
 Frameworks[2].Component[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T2";
 Frameworks[2].Component[1].Class[2].ProtectedSection.Object[1]="r_obj2121";
 Frameworks[2].Component[1].Class[2].ProtectedSection.ObjectType[1]="T1";
 Frameworks[2].Component[1].Class[2].ProtectedSection.Object[2]="r_obj2122";
 Frameworks[2].Component[1].Class[2].ProtectedSection.ObjectType[2]="T2";
 Frameworks[2].Component[1].Class[2].PrivateSection.Method[1].MethodName="v_mthd2121";
 Frameworks[2].Component[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T7";

 Frameworks[2].Component[1].Class[2].PrivateSection.Object[1]="v_obj2121";
 Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[1]="class212";
 Frameworks[2].Component[1].Class[2].PrivateSection.Object[2]="v_obj2122";
 Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[2]="class221";
 Frameworks[2].Component[1].Class[2].PrivateSection.Object[3]="v_obj2123";
 Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[3]="class222";
 Frameworks[2].Component[1].Class[2].PrivateSection.Object[4]="v_obj2124";
 Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[4]="class211";
```

```
//CLASS(2,2,1)
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].MethodName="p_mthd2211";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Z";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="T1";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ReturnedValueType="T1";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[2].MethodName="p_mthd2212";
  Frameworks[2].Component[2].Class[1].PublicSection.Method[2].ReturnedValueType="T1";
  Frameworks[2].Component[2].Class[1].ProtectedSection.Method[1].MethodName="r_mthd2211";
  Frameworks[2].Component[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T5";
Frameworks[2].Component[2].Class[1].ProtectedSection.Object[1]="r_obj2211";
  Frameworks[2].Component[2].Class[1].ProtectedSection.ObjectType[1]="T9";
  Frameworks[2].Component[2].Class[1].ProtectedSection.Object[2]="r_obj2212";
  Frameworks[2].Component[2].Class[1].ProtectedSection.ObjectType[2]="T3";
  Frameworks[2].Component[2].Class[1].PrivateSection.Method[1].MethodName="v_mthd2211";
  Frameworks[2].Component[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T5";
  Frameworks[2].Component[2].Class[1].PrivateSection.Object[1]="v_obj2211";
  Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[1]="T8";
  Frameworks[2].Component[2].Class[1].PrivateSection.Object[2]="v_obj2212";
  Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[2]="T10";
  Frameworks[2].Component[2].Class[1].PrivateSection.Object[3]="v_obj2213";
  Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[3]="class211";
  Frameworks[2].Component[2].Class[1].PrivateSection.Object[4]="v_obj2214";
  Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[4]="class221";//CLASS(2,2,2)
  Frameworks[2].Component[2].Class[2].PublicSection.Method[1].MethodName="p_mthd2221";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ReturnedValueType="T2";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[2].MethodName="p_mthd2222";
  //get ARGS
  Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="T";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T5";
  Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ReturnedValueType="T8";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].MethodName="r_mthd2221";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="R";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T5";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Object[1]="r_obj2221";
  Frameworks[2].Component[2].Class[2].ProtectedSection.ObjectType[1]="T11";
  Frameworks[2].Component[2].Class[2].ProtectedSection.Object[2]="r_obj2222";
  Frameworks[2].Component[2].Class[2].ProtectedSection.ObjectType[2]="T21";
  Frameworks[2].Component[2].Class[2].PrivateSection.Method[1].MethodName="v_mthd2211";
  Frameworks[2].Component[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T5";
  Frameworks[2].Component[2].Class[2].PrivateSection.Object[1]="v_obj2221";
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[1]="T7";
  Frameworks[2].Component[2].Class[2].PrivateSection.Object[2]="v_obj2222";
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[2]="class122";
  Frameworks[2].Component[2].Class[2].PrivateSection.Object[3]="v_obj2223";
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[3]="class121";
  Frameworks[2].Component[2].Class[2].PrivateSection.Object[4]="v_obj2224";
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[4]="class222";//
*****************************************************************
// STEP#4: read in the inheritance references (CLassIDs) for each cless
//*****************************************************************
//Class(1,1,1) inherits from two other classes in public — VIOLation — Multiple inhiretance
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1]=new ClassID();
```

```
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=1;//Class(1,2,1) inherits only from one other class in public

  Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1]=new ClassID();
  Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
  Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=1;//Class(2,1,1) inhirets only from one another class in public
  Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1]=new ClassID();
  Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=1;
  Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;//Class(2,2,1) inherits from one other class in public
  Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1]=new ClassID();
  Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
  Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;
}
protected static void initializeFrameworks_Example_2() {
// ****************************************
// STEP#1 read values for:
//      F, C[], c[][],
//      MP[][][], MR[][][], MV[][][]
//      OP[][][], OR[][][], OV[][][]
// ****************************************
//number of frameworks
  F=3;
//number of components per framework
  C[1]=2; C[2]=2; C[3]=1;
//number of classes per component per framework
  c[1][1]=2; c[1][2]=2;
  c[2][1]=2; c[2][2]=2;
  c[3][1]=1;
//number of public methods per class per component per framework
  MP[1][1][1]=2;  MP[1][1][2]=2;
  MP[1][2][1]=2;  MP[1][2][2]=2;
  MP[2][1][1]=2;  MP[2][1][2]=2;
  MP[2][2][1]=2;  MP[2][2][2]=2;
  MP[3][1][1]=1;

//number of protected methods per class per component per framework
  MR[1][1][1]=1;  MR[1][1][2]=1;
  MR[1][2][1]=1;  MR[1][2][2]=1;
  MR[2][1][1]=1;  MR[2][1][2]=1;
  MR[2][2][1]=1;  MR[2][2][2]=1;
  MR[3][1][1]=1;

//number of private methods per class per component per framework
  MV[1][1][1]=1;    MV[1][1][2]=1;
  MV[1][2][1]=1;    MV[1][2][2]=1;
  MV[2][1][1]=1;    MV[2][1][2]=1;
```

```
  MV[2][2][1]=1;    MV[2][2][2]=1;
  MV[3][1][1]=1;

//number of public objects per class per component per framework
  OP[1][1][1]=0;  OP[1][1][2]=1;
  OP[1][2][1]=1;  OP[1][2][2]=0;
  OP[2][1][1]=1;  OP[2][1][2]=0;
  OP[2][2][1]=0;  OP[2][2][2]=0;
  OP[3][1][1]=1;

//number of protected methods per class per component per framework
  OR[1][1][1]=2;  OR[1][1][2]=2;
  OR[1][2][1]=2;  OR[1][2][2]=2;
  OR[2][1][1]=2;  OR[2][1][2]=2;
  OR[2][2][1]=2;  OR[2][2][2]=2;
  OR[3][1][1]=1;

//number of private methods per class per component per framework
  OV[1][1][1]=4;  OV[1][1][2]=4;
  OV[1][2][1]=4;  OV[1][2][2]=4;
  OV[2][1][1]=4;  OV[2][1][2]=4;
  OV[2][2][1]=4;  OV[2][2][2]=4;
  OV[3][1][1]=1;

//number of classes inherited in public per class per component per framework
  MinhP[1][1][1]=2; MinhP[1][1][2]=0;
  MinhP[1][2][1]=1; MinhP[1][2][2]=0;
  MinhP[2][1][1]=1; MinhP[2][1][2]=0;
  MinhP[2][2][1]=1; MinhP[2][2][2]=0;
  MinhP[3][1][1]=1;

//number of classes inherited in protected per class per component per framework
  MinhR[1][1][1]=1; MinhR[1][1][2]=0;
  MinhR[1][2][1]=0; MinhR[1][2][2]=1;
  MinhR[2][1][1]=0; MinhR[2][1][2]=0;
  MinhR[2][2][1]=0; MinhR[2][2][2]=0;
  MinhR[3][1][1]=1;

//number of classes inherited in private per class per component per framework
  MinhV[1][1][1]=1; MinhV[1][1][2]=0;
  MinhV[1][2][1]=0; MinhV[1][2][2]=0;
  MinhV[2][1][1]=0; MinhV[2][1][2]=0;
  MinhV[2][2][1]=0; MinhV[2][2][2]=0;
  MinhV[3][1][1]=1;
//number of classes inherited from shared pools in public per class per component per framework
  MinhPS[1][1][1]=0;  MinhPS[1][1][2]=0;
  MinhPS[1][2][1]=0;  MinhPS[1][2][2]=0;
  MinhPS[2][1][1]=0;  MinhPS[2][1][2]=0;
  MinhPS[2][2][1]=1;  MinhPS[2][2][2]=0;
  MinhPS[3][1][1]=1;

//number of classes inherited from shared pools in protected per class per component per framework
  MinhRS[1][1][1]=0;  MinhRS[1][1][2]=0;
  MinhRS[1][2][1]=0;  MinhRS[1][2][2]=0;
  MinhRS[2][1][1]=0;  MinhRS[2][1][2]=0;
  MinhRS[2][2][1]=0;  MinhRS[2][2][2]=1;
  MinhRS[3][1][1]=1;
```

```
//number of classes inherited  from shared pools in private per class per component per framework
  MinhVS[1][1][1]=1; MinhVS[1][1][2]=0;
  MinhVS[1][2][1]=0; MinhVS[1][2][2]=0;
  MinhVS[2][1][1]=0; MinhVS[2][1][2]=0;
  MinhVS[2][2][1]=0; MinhVS[2][2][2]=0;
  MinhVS[3][1][1]=1;
//******************************************************
// STEP#2: instantiate for the Frameworks
// objects and the embedied Component, MainSection,
// Class, PublicSection, ProtectedSection, PrivateSection,
// Method, etc. etc, objects
// ALSO: Assign ID values for each object
//******************************************************
// instantiate the framework objects
  for (int i=1; i<=F; i++){
    Frameworks[i]=new FrameworkNode();
    Frameworks[i].FrameworkNodeNumber=i;

    //instantiate the component objects for this new framework object
    for (int j=1; j<=C[i]; j++){
      Frameworks[i].Component[j]=new ComponentNode();
      Frameworks[i].Component[j].ID.ComponentNodeNumber=j;
      Frameworks[i].Component[j].ID.ParentFrameworkNodeNumber=i;
      Frameworks[i].Component[j].MainSection="main_"+i+"_"+j;
      //instantiate the class objects for this component object
      for (int k=1; k<=c[i][j]; k++){
        Frameworks[i].Component[j].Class[k]=new ClassNode();
        Frameworks[i].Component[j].Class[k].ID.ClassNodeNumber=k;
        Frameworks[i].Component[j].Class[k].ID.ClassName="class"+i+j+k;
        Frameworks[i].Component[j].Class[k].ID.ParentComponent.ComponentNodeNumber=j;
        Frameworks[i].Component[j].Class[k].ID.ParentComponent.ParentFrameworkNodeNumber=i;
        //instantiate for the public section of this class
        Frameworks[i].Component[j].Class[k].PublicSection=new PublicSectionNode();
        for (int l=1; l<=MP[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
        //instantiate for the protected section of this class
        Frameworks[i].Component[j].Class[k].ProtectedSection=new ProtectedSectionNode();
        for (int l=1; l<=MR[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
        //instantiate for the private section of this class
```

```
        Frameworks[i].Component[j].Class[k].PrivateSection=new PrivateSectionNode();
        for (int l=1; l<=MV[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
      }
    }
  }
// *****************************************************
// STEP#3: Read the entries for each of these objects
// *****************************************************
//CLASS(1,1,1)
//Class(1,1,1) public section contains only two methods and no objects
//MP[1][1][1]=2 && OP[1][1][1]=0
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].MethodName="p_mthd1111";
  //get in ARGS for this method
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Y";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="T2";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgName[1]="Z";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgType[1]="T3";
//—————————————
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].MethodName="p_mthd1112";
  //get in ARGS for this method
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T3";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
  Frameworks[1].Component[1].Class[1].PublicSection.Method[2].ReturnedValueType="T4";
//Class(1,1,1) protected section contains one method and 2 objects
//MR[1][1][1]=1 && OR[1][1][1]=2
  Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].MethodName="r_mthd1111";
  //get in ARGS for this method
  Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
  Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T2";
  Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgName[1]="Y";
  Frameworks[1].Component[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgType[1]="T3";
//——————————————————————
  Frameworks[1].Component[1].Class[1].ProtectedSection.Object[1]="r_obj1112";
  Frameworks[1].Component[1].Class[1].ProtectedSection.ObjectType[1]="class112";
  Frameworks[1].Component[1].Class[1].ProtectedSection.Object[2]="r_obj1112";
  Frameworks[1].Component[1].Class[1].ProtectedSection.ObjectType[2]="T12";

//Class(1,1,1) private section contains 1 method and 4 objects
//MV[1][1][1]=1 && OV[1][1][1]=4
  Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].MethodName="v_mthd1111";
  //ARGS for this method
  Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgName[1]="Z";
  Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T1";
  Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgName[1]="W";
```

```
  Frameworks[1].Component[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgType[1]="T5";
//————————————————
  Frameworks[1].Component[1].Class[1].PrivateSection.Object[1]="v_obj1111";
  Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[1]="class221";
  Frameworks[1].Component[1].Class[1].PrivateSection.Object[2]="v_obj1112";
  Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[2]="class222";
  Frameworks[1].Component[1].Class[1].PrivateSection.Object[3]="v_objt1113";
  Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[3]="T9";
  Frameworks[1].Component[1].Class[1].PrivateSection.Object[4]="v_obj1114";
  Frameworks[1].Component[1].Class[1].PrivateSection.ObjectType[4]="T5";
//CLASS(1,1,2)
//Class(1,1,2) public section contains 2 methods and 1 object === VIOLATION
//MP[1][1][2]=2 && OP[1][1][2]=1  Frameworks[1].Component[1].Class[2].PublicSection.Method[1].Me-
thodName="p_mthd1121";
  //get ARGS
  Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="A";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[1].ReturnedValueType="T4";
//————————————————
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].MethodName="p_mthd1122";
  //get ARGS
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T3";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
  Frameworks[1].Component[1].Class[2].PublicSection.Method[2].ReturnedValueType="T4";
//————————————————
  Frameworks[1].Component[1].Class[2].PublicSection.Object[1]="p_obj1121";
  Frameworks[1].Component[1].Class[2].PublicSection.ObjectType[1]="T1";
//Class(1,1,2) protected section contains 1 method and 2 objects
//MR[1][1][2]=1 && OR[1][1][2]=2
  Frameworks[1].Component[1].Class[2].ProtectedSection.Method[1].MethodName="r_mthd1121";
  //get ARGS
  Frameworks[1].Component[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
//————————————————
  Frameworks[1].Component[1].Class[2].ProtectedSection.Object[1]="r_obj1121";
  Frameworks[1].Component[1].Class[2].ProtectedSection.ObjectType[1]="class212";
  Frameworks[1].Component[1].Class[2].ProtectedSection.Object[2]="r_obj1122";
  Frameworks[1].Component[1].Class[2].ProtectedSection.ObjectType[2]="T5";
//Class(1,1,2) private section contains 1 method and 4 objects
//MV[1][1][2]=1 && OV[1][1][2]=4
  Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].MethodName="v_mthd1121";
  //get ARGS
  Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ARGS.IN_ArgName[1]="A";
  Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T3";
  Frameworks[1].Component[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T3";
//————————————————
  Frameworks[1].Component[1].Class[2].PrivateSection.Object[1]="v_obj1121";
  Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[1]="class111";
  Frameworks[1].Component[1].Class[2].PrivateSection.Object[2]="v_obj1122";
  Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[2]="T6";
  Frameworks[1].Component[1].Class[2].PrivateSection.Object[3]="v_obj1123";
  Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[3]="T7";
  Frameworks[1].Component[1].Class[2].PrivateSection.Object[4]="v_obj1124";
  Frameworks[1].Component[1].Class[2].PrivateSection.ObjectType[4]="T1";
//CLASS(1,2,1)
//Class(1,2,1) public section contains 2 methods and 1 object === VIOLATION
//MP[1][2][1]=2 && OP[1][2][1]=1
```

```
  Frameworks[1].Component[2].Class[1].PublicSection.Method[1].MethodName="p_mthd1211";
  // this method has no ARGS
//————————————————
  Frameworks[1].Component[2].Class[1].PublicSection.Method[2].MethodName="p_mthd1212";
  //get ARGS
  Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="X";
  Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
  Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.OUT_ArgName[1]="Z";
  Frameworks[1].Component[2].Class[1].PublicSection.Method[2].ARGS.OUT_ArgType[1]="T3";
//————————————————————————
  Frameworks[1].Component[2].Class[1].PublicSection.Object[1]="pub_object1211";
  Frameworks[1].Component[2].Class[1].PublicSection.ObjectType[1]="T11";
//Class(1,2,1) protected section contains 1 method and 2 objects
//MR[1][2][1]=1 && OR[1][2][1]=2
  Frameworks[1].Component[2].Class[1].ProtectedSection.Method[1].MethodName="r_mthd1211";
  //get ARGS
  Frameworks[1].Component[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T3";
//————————————————
  Frameworks[1].Component[2].Class[1].ProtectedSection.Object[1]="r_obj1211";
  Frameworks[1].Component[2].Class[1].ProtectedSection.ObjectType[1]="class212";
  Frameworks[1].Component[2].Class[1].ProtectedSection.Object[2]="r_obj1212";
  Frameworks[1].Component[2].Class[1].ProtectedSection.ObjectType[2]="T10";
//Class(1,2,1) private section contains 1 method and 4 objects
//MV[1][2][1]=1 && OV[1][2][1]=4
  Frameworks[1].Component[2].Class[1].PrivateSection.Method[1].MethodName="v_mthd1211";
  //get ARGS
  Frameworks[1].Component[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
//————————————————
  Frameworks[1].Component[2].Class[1].PrivateSection.Object[1]="v_obj1211";
  Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[1]="class212";
  Frameworks[1].Component[2].Class[1].PrivateSection.Object[2]="v_obj1212";
  Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[2]="T6";
  Frameworks[1].Component[2].Class[1].PrivateSection.Object[3]="v_obj1213";
  Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[3]="T2";
  Frameworks[1].Component[2].Class[1].PrivateSection.Object[4]="v_obj1214";
  Frameworks[1].Component[2].Class[1].PrivateSection.ObjectType[4]="class222";
    //CLASS(1,2,2)
//Class(1,2,2) public section contains 2 methods and no objects
//MP[1][2][2]=2 && OP[1][2][2]=0
  Frameworks[1].Component[2].Class[2].PublicSection.Method[1].MethodName="p_mthd1221";
  //get ARGS
  Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="Y";
  Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T2";
  Frameworks[1].Component[2].Class[2].PublicSection.Method[1].ReturnedValueType="T2";
//————————————————
  Frameworks[1].Component[2].Class[2].PublicSection.Method[2].MethodName="p_mthd1222";
  //get ARGS
  Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="Y";
  Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
  Frameworks[1].Component[2].Class[2].PublicSection.Method[2].ReturnedValueType="T1";
//———————————————— //Class(1,2,2) protected section contains 1 method and 2 objects
//MR[1][2][2]=1 && OR[1][2][2]=2
  Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].MethodName="r_mthd1221";
  //get ARGS
  Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
  Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T4";
  Frameworks[1].Component[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T1";
//————————————————
```

```
  Frameworks[1].Component[2].Class[2].ProtectedSection.Object[1]="r_obj1221";
  Frameworks[1].Component[2].Class[2].ProtectedSection.ObjectType[1]="T11";
  Frameworks[1].Component[2].Class[2].ProtectedSection.Object[2]="r_obj1222";
  Frameworks[1].Component[2].Class[2].ProtectedSection.ObjectType[2]="T6";
//Class(1,2,2) private section contains 1 method and 4 objects
//MV[1][2][2]=1 && OV[1][2][2]=4
  Frameworks[1].Component[2].Class[2].PrivateSection.Method[1].MethodName="v_mthd1221";
  //get ARGS
  Frameworks[1].Component[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T1";
//-----------------------------------------
  Frameworks[1].Component[2].Class[2].PrivateSection.Object[1]="v_obj1221";
  Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[1]="class111";
  Frameworks[1].Component[2].Class[2].PrivateSection.Object[2]="v_obj1222";
  Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[2]="class221";
  Frameworks[1].Component[2].Class[2].PrivateSection.Object[3]="v_obj1223";
  Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[3]="T12";
  Frameworks[1].Component[2].Class[2].PrivateSection.Object[4]="v_obj1224";
  Frameworks[1].Component[2].Class[2].PrivateSection.ObjectType[4]="T8";

//CLASS(2,1,1)
//Class(2,1,1) public section contains 2 methods and 1 object — VIOLATION
//MP[2][1][1]=2 && OP[2][1][1]=1
  Frameworks[2].Component[1].Class[1].PublicSection.Method[1].MethodName="p_mthd2111";
// get ARGS
  Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
  Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T6";
  Frameworks[2].Component[1].Class[1].PublicSection.Method[1].ReturnedValueType="T3";
//-----------------------------------------
  Frameworks[2].Component[1].Class[1].PublicSection.Method[2].MethodName="p_mthd2112";
// get ARGS
  Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="W";
  Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
  Frameworks[2].Component[1].Class[1].PublicSection.Method[2].ReturnedValueType="T2";
//-----------------------------------------
  Frameworks[2].Component[1].Class[1].PublicSection.Object[1]="p_obj2111";
  Frameworks[2].Component[1].Class[1].PublicSection.ObjectType[1]="T12";
//Class(2,1,1) protected section contains 1 method and 2 objects
//MR[2][1][1]=1 && OR[2][1][1]=2
  Frameworks[2].Component[1].Class[1].ProtectedSection.Method[1].MethodName="r_mthd2111";
// get ARGS
  Frameworks[2].Component[1].Class[1].ProtectedSection.Method[1].ReturnedValueType="T1";
//-----------------------------------------
  Frameworks[2].Component[1].Class[1].ProtectedSection.Object[1]="r_obj2111";
  Frameworks[2].Component[1].Class[1].ProtectedSection.ObjectType[1]="class112";
  Frameworks[2].Component[1].Class[1].ProtectedSection.Object[2]="r_obj2112";
  Frameworks[2].Component[1].Class[1].ProtectedSection.ObjectType[2]="T13";
//Class(2,1,1) private section contains 1 method and 4 objects
//MV[2][1][1]=1 && OV[2][1][1]=4
  Frameworks[2].Component[1].Class[1].PrivateSection.Method[1].MethodName="v_mthd2111";
// get ARGS
  Frameworks[2].Component[1].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
//-----------------------------------------
  Frameworks[2].Component[1].Class[1].PrivateSection.Object[1]="v_obj2111";
  Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[1]="class121";
  Frameworks[2].Component[1].Class[1].PrivateSection.Object[2]="v_obj2112";
  Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[2]="class122";
  Frameworks[2].Component[1].Class[1].PrivateSection.Object[3]="v_obj2113";
  Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[3]="T7";
```

```
Frameworks[2].Component[1].Class[1].PrivateSection.Object[4]="v_obj2114";
Frameworks[2].Component[1].Class[1].PrivateSection.ObjectType[4]="T6";
//CLASS(2,1,2)
//Class(2,1,2) public section contains 2 methods and no object
//MP[2][1][2]=2 && OP[2][1][2]=0
Frameworks[2].Component[1].Class[2].PublicSection.Method[1].MethodName="p_mthd2121";
// get ARGS
Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
Frameworks[2].Component[1].Class[2].PublicSection.Method[1].ReturnedValueType="T1";
//————————————————
Frameworks[2].Component[1].Class[2].PublicSection.Method[2].MethodName="p_mthd2122";
// get ARGS
Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="W";
Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T2";
Frameworks[2].Component[1].Class[2].PublicSection.Method[2].ReturnedValueType="T3";
//————————————————//Class(2,1,2) protected section contains 1 method and 2 objects
//MR[2][1][2]=1 && OR[2][1][2]=2
Frameworks[2].Component[1].Class[2].ProtectedSection.Method[1].MethodName="r_mthd2121";
// get ARGS
Frameworks[2].Component[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T2";
//————————————————
Frameworks[2].Component[1].Class[2].ProtectedSection.Object[1]="r_obj2121";
Frameworks[2].Component[1].Class[2].ProtectedSection.ObjectType[1]="T1";
Frameworks[2].Component[1].Class[2].ProtectedSection.Object[2]="r_obj2122";
Frameworks[2].Component[1].Class[2].ProtectedSection.ObjectType[2]="T2";
//Class(2,1,2) private section contains 1 method and 4 objects
//MV[2][1][2]=1 && OV[2][1][2]=4
Frameworks[2].Component[1].Class[2].PrivateSection.Method[1].MethodName="v_mthd2121";
// get ARGS
Frameworks[2].Component[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T7";
//————————————————
Frameworks[2].Component[1].Class[2].PrivateSection.Object[1]="v_obj2121";
Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[1]="class212";
Frameworks[2].Component[1].Class[2].PrivateSection.Object[2]="v_obj2122";
Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[2]="class221";
Frameworks[2].Component[1].Class[2].PrivateSection.Object[3]="v_obj2123";
Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[3]="class222";
Frameworks[2].Component[1].Class[2].PrivateSection.Object[4]="v_obj2124";
Frameworks[2].Component[1].Class[2].PrivateSection.ObjectType[4]="class211";
//CLASS(2,2,1)
//Class(2,2,1) public section contains 2 methods and no object
//MP[2][2][1]=2 && OP[2][2][1]=0
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].MethodName="p_mthd2211";
// get ARGS
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="Z";
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T1";
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Z";
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="T1";
Frameworks[2].Component[2].Class[1].PublicSection.Method[1].ReturnedValueType="T1";
//————————————————
Frameworks[2].Component[2].Class[1].PublicSection.Method[2].MethodName="p_mthd2212";
//get ARGS
Frameworks[2].Component[2].Class[1].PublicSection.Method[2].ReturnedValueType="T1";
//————————————————//Class(2,2,1) protected section contains 1 method and 2 ob-
jects
//MR[2][2][1]=1 && OR[2][2][1]=2
Frameworks[2].Component[2].Class[1].ProtectedSection.Method[1].MethodName="r_mthd2211";
```

```
//get ARGS
 Frameworks[2].Component[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T5";
//———————————
 Frameworks[2].Component[2].Class[1].ProtectedSection.Object[1]="r_obj2211";
 Frameworks[2].Component[2].Class[1].ProtectedSection.ObjectType[1]="T9";
 Frameworks[2].Component[2].Class[1].ProtectedSection.Object[2]="r_obj2212";
 Frameworks[2].Component[2].Class[1].ProtectedSection.ObjectType[2]="T3";
//Class(2,2,1) private section contains 1 method and 4 objects
//MV[2][2][1]=1 && OV[2][2][1]=4
 Frameworks[2].Component[2].Class[1].PrivateSection.Method[1].MethodName="v_mthd2211";
 //get ARGS
 Frameworks[2].Component[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T5";
//———————————
 Frameworks[2].Component[2].Class[1].PrivateSection.Object[1]="v_obj2211";
 Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[1]="T8";
 Frameworks[2].Component[2].Class[1].PrivateSection.Object[2]="v_obj2212";
 Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[2]="T10";
 Frameworks[2].Component[2].Class[1].PrivateSection.Object[3]="v_obj2213";
 Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[3]="class211";
 Frameworks[2].Component[2].Class[1].PrivateSection.Object[4]="v_obj2214";
 Frameworks[2].Component[2].Class[1].PrivateSection.ObjectType[4]="class221";
//CLASS(2,2,2)
 //Class(2,2,2) public section contains 2 methods and no object
 //MP[2][2][2]=2 && OP[2][2][1]=0
 Frameworks[2].Component[2].Class[2].PublicSection.Method[1].MethodName="p_mthd2221";
 //get ARGS
 Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
 Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
 Frameworks[2].Component[2].Class[2].PublicSection.Method[1].ReturnedValueType="T2";
//———————————
 Frameworks[2].Component[2].Class[2].PublicSection.Method[2].MethodName="p_mthd2222";
 //get ARGS
 Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="T";
 Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T5";
 Frameworks[2].Component[2].Class[2].PublicSection.Method[2].ReturnedValueType="T8";
//———————————————————————//Class(2,2,2) protected section contains 1 method and 2 objects
//MR[2][2][2]=1 && OR[2][2][2]=2
 Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].MethodName="r_mthd2221";
 //get ARGS
 Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="R";
 Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T5";
 Frameworks[2].Component[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
//———————————
 Frameworks[2].Component[2].Class[2].ProtectedSection.Object[1]="r_obj2221";
 Frameworks[2].Component[2].Class[2].ProtectedSection.ObjectType[1]="T11";
 Frameworks[2].Component[2].Class[2].ProtectedSection.Object[2]="r_obj2222";
 Frameworks[2].Component[2].Class[2].ProtectedSection.ObjectType[2]="T21";
//Class(2,2,2) private section contains 1 method and 4 objects
//MV[2][2][2]=1 && OV[2][2][2]=4
 Frameworks[2].Component[2].Class[2].PrivateSection.Method[1].MethodName="v_mthd2211";
 //get ARGS
 Frameworks[2].Component[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T5";
//———————————
 Frameworks[2].Component[2].Class[2].PrivateSection.Object[1]="v_obj2221";
 Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[1]="T7";
 Frameworks[2].Component[2].Class[2].PrivateSection.Object[2]="v_obj2222";
 Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[2]="class122";
 Frameworks[2].Component[2].Class[2].PrivateSection.Object[3]="v_obj2223";
```

```
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[3]="class121";
  Frameworks[2].Component[2].Class[2].PrivateSection.Object[4]="v_obj2224";
  Frameworks[2].Component[2].Class[2].PrivateSection.ObjectType[4]="class222";
//CLASS(3,1,1)
//Class(3,1,1) public section contains 1 methods and 1 object === VIOLATION
//MP[3][1][1]=1 && OP[3][1][1]=1
  Frameworks[3].Component[1].Class[1].PublicSection.Method[1].MethodName="p_mthd3111";
  //get ARGS
  Frameworks[3].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="W";
  Frameworks[3].Component[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="T4";
  Frameworks[3].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgName[1]="Z";
  Frameworks[3].Component[1].Class[1].PublicSection.Method[1].ARGS.OUT_ArgType[1]="T3";
//————————————————————————
  Frameworks[3].Component[1].Class[1].PublicSection.Object[1]="pub_object3111";
  Frameworks[3].Component[1].Class[1].PublicSection.ObjectType[1]="class121";
//Class(3,1,1) protected section contains 1 method and 1 objects
//MR[3][1][1]=1 && OR[3][1][1]=1
  Frameworks[3].Component[1].Class[1].ProtectedSection.Method[1].MethodName="r_mthd3111";
  //get ARGS
  Frameworks[3].Component[1].Class[1].ProtectedSection.Method[1].ReturnedValueType="T1";
//————————————————
  Frameworks[3].Component[1].Class[1].ProtectedSection.Object[1]="r_obj3111";
  Frameworks[3].Component[1].Class[1].ProtectedSection.ObjectType[1]="class212";
//Class(3,1,1) private section contains 1 method and 1 objects
//MV[3][1][1]=1 && OV[3][1][1]=1
  Frameworks[3].Component[1].Class[1].PrivateSection.Method[1].MethodName="v_mthd3111";
  //get ARGS
  Frameworks[3].Component[1].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
//————————————————
  Frameworks[3].Component[1].Class[1].PrivateSection.Object[1]="v_obj3111";
  Frameworks[3].Component[1].Class[1].PrivateSection.ObjectType[1]="class122";
//*******************************************************************
// STEP#4: read in the inheritance references (CLassIDs) for each cless
//*******************************************************************
//Class(1,1,1) inherits from two other classes in public — VIOLation — Multiple inhiretance
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1]=new ClassID();
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=2;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=1;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[2]=new ClassID();
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[2].ClassNodeNumber=1;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[2].ParentComponent.ComponentNodeNum-
ber=2;
  Frameworks[1].Component[1].Class[1].PublicInheritedClasses[2].ParentComponent.ParentFrameworkNo-
deNumber=2;

//Class(1,1,1) inherits another class in protected — Violation — non-public inheritance
  Frameworks[1].Component[1].Class[1].ProtectedInheritedClasses[1]=new ClassID();
  Frameworks[1].Component[1].Class[1].ProtectedInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[1].Component[1].Class[1].ProtectedInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[1].Component[1].Class[1].ProtectedInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=2;// Class(1,1,1) inherits from another class in private — Violation — non-public inheritance
  Frameworks[1].Component[1].Class[1].PrivateInheritedClasses[1]=new ClassID();
  Frameworks[1].Component[1].Class[1].PrivateInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[1].Component[1].Class[1].PrivateInheritedClasses[1].ParentComponent.ComponentNode-
```

Number=1;
Frameworks[1].Component[1].Class[1].PrivateInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;// Class(1,1,1) inherits another class in private from shared pools
Frameworks[1].Component[1].Class[1].PrivateInheritedSharedClasses[1]=new SharedClassID();
Frameworks[1].Component[1].Class[1].PrivateInheritedSharedClasses[1].ClassNodeNumber=1;
Frameworks[1].Component[1].Class[1].PrivateInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;
// Class(1,1,2) does not inherit from any other class//Class(1,2,1) inherits only from one other class in public

Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1]=new ClassID();
Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
Frameworks[1].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=1;//Class(1,2,2) inhirets only from one another class in protected
Frameworks[1].Component[2].Class[2].ProtectedInheritedClasses[1]=new ClassID();
Frameworks[1].Component[2].Class[2].ProtectedInheritedClasses[1].ClassNodeNumber=1;
Frameworks[1].Component[2].Class[2].ProtectedInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
Frameworks[1].Component[2].Class[2].ProtectedInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;//Class(2,1,1) inhirets only from one another class in public
Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1]=new ClassID();
Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=1;
Frameworks[2].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;//Class(2,1,2) does not inherit from any other class//Class(2,2,1) inherits from one other class in public
Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1]=new ClassID();
Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
Frameworks[2].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;
//Class(2,2,1) inherits in public another class from shared pools
Frameworks[2].Component[2].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
Frameworks[2].Component[2].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=1;
Frameworks[2].Component[2].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;
//Class(2,2,2) does not inherit from another class from any framework
//Class(2,2,2) inherits in protected another class from shared pools
Frameworks[2].Component[2].Class[2].ProtectedInheritedSharedClasses[1]=new SharedClassID();
Frameworks[2].Component[2].Class[2].ProtectedInheritedSharedClasses[1].ClassNodeNumber=2;
Frameworks[2].Component[2].Class[2].ProtectedInheritedSharedClasses[1].ParentSharedPoolNodeNumber=1;

//Class(3,1,1) inherits from two other classes in public — VIOLation — Multiple inhiretance
Frameworks[3].Component[1].Class[1].PublicInheritedClasses[1]=new ClassID();
Frameworks[3].Component[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[3].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
Frameworks[3].Component[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=2;
// Class(3,1,1) inherits another class in public from shared pools
Frameworks[3].Component[1].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
Frameworks[3].Component[1].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=1;
Frameworks[3].Component[1].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=1;

//Class(3,1,1) inherits another class in protected — Violation — non–public inheritance

```
  Frameworks[3].Component[1].Class[1].ProtectedInheritedClasses[1]=new ClassID();
  Frameworks[3].Component[1].Class[1].ProtectedInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[3].Component[1].Class[1].ProtectedInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[3].Component[1].Class[1].ProtectedInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=1;
// Class(3,1,1) inherits another class in protected from shared pools
  Frameworks[3].Component[1].Class[1].ProtectedInheritedSharedClasses[1]=new SharedClassID();
  Frameworks[3].Component[1].Class[1].ProtectedInheritedSharedClasses[1].ClassNodeNumber=2;
  Frameworks[3].Component[1].Class[1].ProtectedInheritedSharedClasses[1].ParentSharedPoolNodeNum-
ber=2;
// Class(3,1,1) inherits from another class in private — Violation — non-public inheritance
  Frameworks[3].Component[1].Class[1].PrivateInheritedClasses[1]=new ClassID();
  Frameworks[3].Component[1].Class[1].PrivateInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[3].Component[1].Class[1].PrivateInheritedClasses[1].ParentComponent.ComponentNode-
Number=2;
  Frameworks[3].Component[1].Class[1].PrivateInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=2;
// Class(3,1,1) inherits another class in private from shared pools
  Frameworks[3].Component[1].Class[1].PrivateInheritedSharedClasses[1]=new SharedClassID();
  Frameworks[3].Component[1].Class[1].PrivateInheritedSharedClasses[1].ClassNodeNumber=2;
  Frameworks[3].Component[1].Class[1].PrivateInheritedSharedClasses[1].ParentSharedPoolNodeNum-
ber=1;
}
protected static void initializeFrameworks_Example_3_ACE() {
//****************************************
// STEP#1 read values for:
//      F, C[], c[][],
//      MP[][][], MR[][][], MV[][][]
//      OP[][][], OR[][][], OV[][][]
//****************************************
//number of frameworks
  F=15;
//number of components per framework
  C[1]=9; // [ASX]
  C[2]=2; // [CORBA]
  C[3]=3; // [Collections]
  C[4]=9; // [Concurrency]
  C[5]=5; // [Connection]
  C[6]=44; // [IPC]
  C[7]=2; // [Log_Msg]
  C[8]=7; // [Memory]
  C[9]=4; // [Misc]
  C[10]=7; // [Name_Service]
  C[11]=1; // [OS]
  C[12]=8; // [Reactor]
  C[13]=7; // [System_V_IPC]
  C[14]=2; // [Timers]
  C[15]=7;
//number of classes per component per framework
  c[1][1]=1; c[1][2]=3; c[1][3]=1; c[1][4]=4; c[1][5]=1;
        c[1][6]=2; c[1][7]=2; c[1][8]=3; c[1][9]=3;

  c[2][1]=3; c[2][2]=1;

  c[3][1]=10; c[3][2]=8; c[3][3]=4;

  c[4][1]=1;
```

```
c[4][2]=1;
c[4][3]=4;
c[4][4]=23;
c[4][5]=1;
c[4][6]=14;
c[4][7]=1;
c[4][8]=6;
c[4][9]=1; c[5][1]=3; c[5][2]=2; c[5][3]=1; c[5][4]=5; c[5][5]=1;
c[6][1]=1; c[6][2]=1; c[6][3]=1; c[6][4]=1; c[6][5]=1;
c[6][6]=1; c[6][7]=1; c[6][8]=1; c[6][9]=1; c[6][10]=1;
c[6][11]=1; c[6][12]=1; c[6][13]=1; c[6][14]=1; c[6][15]=1;
c[6][16]=1; c[6][17]=1; c[6][18]=1; c[6][19]=1; c[6][20]=1;
c[6][21]=1; c[6][22]=1; c[6][23]=1; c[6][24]=2; c[6][25]=1;
c[6][26]=1; c[6][27]=1; c[6][28]=1; c[6][29]=1; c[6][30]=1;
c[6][31]=1; c[6][32]=1; c[6][33]=1; c[6][34]=1; c[6][35]=1;
c[6][36]=1; c[6][37]=1; c[6][38]=1; c[6][39]=3; c[6][40]=1;
c[6][41]=1; c[6][42]=1; c[6][43]=1; c[6][44]=1;   c[7][1]=1; c[7][2]=1;
c[8][1]=1; c[8][2]=3; c[8][3]=3; c[8][4]=9; c[8][5]=1;
c[8][6]=1; c[8][7]=1; c[9][1]=2; c[9][2]=1; c[9][3]=1; c[9][4]=1;

c[10][1]=2; c[10][2]=2; c[10][3]=1; c[10][4]=2; c[10][5]=2;
c[10][6]=2; c[10][7]=1;

c[11][1]=5; c[12][1]=1; c[12][2]=1; c[12][3]=2; c[12][4]=8; c[12][5]=3;
c[12][6]=7; c[12][7]=2; c[12][8]=1;

c[13][1]=1; c[13][2]=1; c[13][3]=1; c[13][4]=1; c[13][5]=1;
c[13][6]=1; c[13][7]=1; c[14][1]=2; c[14][2]=1; c[15][1]=15;   c[15][2]=1;   c[15][3]=5;   c[15][4]=1;
c[15][5]=1;
c[15][6]=3; c[15][7]=2;// class names
// [ASX]
className[1][1][1]=  "ACE_Module_Link";
className[1][2][1]=  "ACE_Map_Manager";
className[1][2][2]=  "ACE_Map_Iterator";
className[1][2][3]=  "ACE_Map_Reverse_Iterator";
className[1][3][1]=  "ACE_Message_Block";
className[1][4][1]=  "ACE_Notification_Strategy";
className[1][4][2]=  "ACE_Message_Queue";
className[1][4][3]=  "ACE_Message_Queue_Iterator";
className[1][4][4]=  "ACE_Message_Queue_Reverse_Iterator";
className[1][5][1]=  "ACE_Module";
className[1][6][1]=  "Driver";
className[1][6][2]=  "ACE_Multiplexor";
className[1][7][1]=  "ACE_Stream";
className[1][7][2]=  "ACE_Stream_Iterator";
className[1][8][1]=  "ACE_Stream_Head";
className[1][8][2]=  "ACE_Stream_Tail";
className[1][8][3]=  "ACE_Thru_Task";
className[1][9][1]=  "ACE_Task_Flags";
className[1][9][2]=  "ACE_Task_Base";
className[1][9][3]=  "ACE_Task_Exit";// [CORBA]
className[2][1][1]=  "ACE_CORBA_Handler";
className[2][1][2]=  "ACE_ST_CORBA_Handler";
className[2][1][3]=  "ACE_MT_CORBA_Handler";
className[2][2][1]=  "ACE_CORBA_Ref";// [Collections]
className[3][1][1]=  "ACE_Set_Node";
className[3][1][2]=  "ACE_Unbounded_Set_Iterator";
className[3][1][3]=  "ACE_Set_Node";
```

```
className[3][1][4]=  "ACE_Unbounded_Set";
className[3][1][5]=  "ACE_Fixed_Set";
className[3][1][6]=  "ACE_Fixed_Set_Iterator";
className[3][1][7]=  "ACE_Fixed_Set";
className[3][1][8]=  "ACE_Bounded_Set";
className[3][1][9]=  "ACE_Bounded_Set_Iterator";
className[3][1][10]= "ACE_Bounded_Set";
className[3][2][1]=  "ACE_Stack_Node";
className[3][2][2]=  "ACE_Queue_Node";
className[3][2][3]=  "ACE_Bounded_Stack";
className[3][2][4]=  "ACE_Fixed_Stack";
className[3][2][5]=  "ACE_Stack_Node";
className[3][2][6]=  "ACE_Unbounded_Stack ";
className[3][2][7]=  "ACE_Queue_Node";
className[3][2][8]=  "ACE_Unbounded_Queue";
className[3][3][1]=  "ACE_Allocator";
className[3][3][2]=  "ACE_CString";
className[3][3][3]=  "ACE_SString";
className[3][3][4]=  "ACE_WString";// [Concurrency]
className[4][1][1]=  "ACE_Activation_Queue";
className[4][2][1]=  "ACE_Method_Object ";
className[4][3][1]=  "ACE_Process_Descriptor";
className[4][3][2]=  "ACE_Process_Control";
className[4][3][3]=  "ACE_Process_Manager";
className[4][3][4]=  "ACE_Process_Control";
className[4][4][1]=  "ACE_File_Lock";
className[4][4][2]=  "ACE_Semaphore";
className[4][4][3]=  "ACE_Process_Semaphore";
className[4][4][4]=  "ACE_RW_Mutex";
className[4][4][5]=  "ACE_Mutex";
className[4][4][6]=  "ACE_Process_Mutex";
className[4][4][7]=  "ACE_RW_Process_Mutex";
className[4][4][8]=  "ACE_Null_Barrier";
className[4][4][9]=  "ACE_Null_Mutex";
className[4][4][10]= "ACE_Null_Condition_Mutex";
className[4][4][11]= "ACE_Null_Mutex_Guard";
className[4][4][12]= "ACE_Event";
className[4][4][13]= "ACE_Manual_Event";
className[4][4][14]= "ACE_Auto_Event";
className[4][4][15]= "ACE_Thread_Mutex";
className[4][4][16]= "ACE_Thread_Mutex_Guard";
className[4][4][17]= "ACE_Condition_Thread_Mutex";
className[4][4][18]= "ACE_Recursive_Thread_Mutex";
className[4][4][19]= "ACE_RW_Thread_Mutex";
className[4][4][20]= "ACE_Thread_Semaphore";
className[4][4][21]= "ACE_Barrier";
className[4][4][22]= "ACE_Process_Barrier";
className[4][4][23]= "ACE_Thread_Barrier";
className[4][5][1]=  "ACE_Synch_Options";
className[4][6][1]=  "ACE_Test_and_Set";
className[4][6][2]=  "ACE_Atomic_Op";
className[4][6][3]=  "ACE_TSS";
className[4][6][4]=  "ACE_NULL_SYNCH";
className[4][6][5]=  "ACE_Guard";
className[4][6][6]=  "ACE_Write_Guard";
className[4][6][7]=  "ACE_Read_Guard";
className[4][6][8]=  "ACE_TSS_Guard";
className[4][6][9]=  "ACE_TSS_Write_Guard";
```

```
className[4][6][10]= "ACE_TSS_Read_Guard";
className[4][6][11]= "ACE_Condition";
className[4][6][12]= "ACE_Process_Condition";
className[4][6][13]= "ACE_Thread_Condition";
className[4][6][14]= "ACE_MT_SYNCH";
className[4][7][1]=  "ACE_Thread";
className[4][8][1]=  "ACE_Task_Base";
className[4][8][2]=  "ACE_Thread_Manager";
className[4][8][3]=  "ACE_Thread_Descriptor";
className[4][8][4]=  "ACE_Thread_Control";
className[4][8][5]=  "ACE_Thread_Manager";
className[4][8][6]=  "ACE_Thread_Control";
className[4][9][1]=  "ACE_Token";// [Connection]
className[5][1][1]=  "ACE_Acceptor";
className[5][1][2]=  "ACE_Strategy_Acceptor";
className[5][1][3]=  "ACE_Oneshot_Acceptor";
className[5][2][1]=  "ACE_Svc_Tuple";
className[5][2][2]=  "ACE_Connector";
className[5][3][1]=  "ACE_Dynamic_Service";
className[5][4][1]=  "ACE_Reactor";
className[5][4][2]=  "ACE_ReactorEx";
className[5][4][3]=  "ACE_Notification_Strategy";
className[5][4][4]=  "ACE_Reactor_Notification_Strategy";
className[5][4][5]=  "ACE_ReactorEx_Notification_Strategy";
className[5][5][1]=  "ACE_Svc_Handler";// [IPC]
className[6][1][1]=  "ACE_IO_SAP";
className[6][2][1]=  "ACE_DEV";
className[6][3][1]=  "ACE_DEV_Connector";
className[6][4][1]=  "ACE_DEV_IO";
className[6][5][1]=  "ACE_FILE";
className[6][6][1]=  "ACE_FILE_Connector";
className[6][7][1]=  "ACE_FILE_IO";
className[6][8][1]=  "ACE_IPC_SAP";
className[6][9][1]=  "ACE_Addr";
className[6][10][1]= "ACE_DEV_Addr";
className[6][11][1]= "ACE_FILE_Addr";
className[6][12][1]= "ACE_INET_Addr";
className[6][13][1]= "ACE_SPIPE_Addr";
className[6][14][1]= "ACE_UNIX_Addr";
className[6][15][1]= "ACE_FIFO";
className[6][16][1]= "ACE_FIFO_Recv";
className[6][17][1]= "ACE_FIFO_Recv_Msg";
className[6][18][1]= "ACE_FIFO_Send";
className[6][19][1]= "ACE_FIFO_Send_Msg";
className[6][20][1]= "ACE_LSOCK";
className[6][21][1]= "ACE_LSOCK_Acceptor";
className[6][22][1]= "ACE_LSOCK_CODgram";
className[6][23][1]= "ACE_LSOCK_Connector";
className[6][24][1]= "ACE_LSOCK_Dgram";
className[6][24][2]= "ACE_LSOCK_Stream";
className[6][25][1]= "ACE_SOCK";
className[6][26][1]= "ACE_SOCK_Acceptor";
className[6][27][1]= "ACE_SOCK_CODgram";
className[6][28][1]= "ACE_SOCK_Connector";
className[6][29][1]= "ACE_SOCK_Dgram";
className[6][30][1]= "ACE_SOCK_Dgram_Bcast";
className[6][31][1]= "ACE_SOCK_Dgram_Mcast";
className[6][32][1]= "ACE_SOCK_IO";
```

```
className[6][33][1]= "ACE_SOCK_Stream";
className[6][34][1]= "ACE_SPIPE";
className[6][35][1]= "ACE_SPIPE_Acceptor";
className[6][36][1]= "ACE_SPIPE_Connector";
className[6][37][1]= "ACE_SPIPE_Stream";
className[6][38][1]= "ACE_TLI";
className[6][39][1]= "ACE_TLI_Request_Queue";
className[6][39][2]= "ACE_TLI_Request_Queue";
className[6][39][3]= "ACE_TLI_Acceptor";
className[6][40][1]= "ACE_TLI_Connector";
className[6][41][1]= "ACE_TLI_Stream";
className[6][42][1]= "ACE_UPIPE_Acceptor";
className[6][43][1]= "ACE_UPIPE_Connector";
className[6][44][1]= "ACE_UPIPE_Stream";// [Log_Msg]
className[7][1][1]=  "ACE_Log_Msg ";
className[7][2][1]=  "ACE_Log_Record";// [Memory]
className[8][1][1]=  "ACE_Mem_Map";
className[8][2][1]=  "ACE_Allocator";
className[8][2][2]=  "ACE_Name_Node";
className[8][2][3]=  "ACE_Control_Block";
className[8][3][1]=  "ACE_Allocator_Adapter";
className[8][3][2]=  "ACE_Malloc";
className[8][3][3]=  "ACE_Malloc_Iterator";
className[8][4][1]=  "ACE_Sbrk_Memory_Pool_Options";
className[8][4][2]=  "ACE_Sbrk_Memory_Pool";
className[8][4][3]=  "ACE_Shared_Memory_Pool_Options";
className[8][4][4]=  "ACE_Shared_Memory_Pool";
className[8][4][5]=  "ACE_Local_Memory_Pool_Options";
className[8][4][6]=  "ACE_Local_Memory_Pool";
className[8][4][7]=  "ACE_MMAP_Memory_Pool_Options";
className[8][4][8]=  "ACE_MMAP_Memory_Pool";
className[8][4][9]=  "ACE_Lite_MMAP_Memory_Pool";
className[8][5][1]=  "ACE_Shared_Memory";
className[8][6][1]=  "ACE_Shared_Memory_MM";
className[8][7][1]=  "ACE_Shared_Memory_SV";// [Misc]
className[9][1][1]=  "auto_ptr";
className[9][1][2]=  "auto_array_ptr";
className[9][2][1]=  "ACE_Dynamic";
className[9][3][1]=  "ACE_Get_Opt";
className[9][4][1]=  "ACE_Singleton";// [Name_Service]
className[10][1][1]= "ACE_NS_String";
className[10][1][2]= "ACE_NS_Internal";
className[10][2][1]= "ACE_Name_Space_Map";
className[10][2][2]= "ACE_Local_Name_Space";
className[10][3][1]= "ACE_Name_Proxy";
className[10][4][1]= "ACE_Name_Request";
className[10][4][2]= "ACE_Name_Reply";
className[10][5][1]= "ACE_Name_Binding";
className[10][5][2]= "ACE_Name_Space";
className[10][6][1]= "ACE_Naming_Context";
className[10][6][2]= "ACE_Name_Options";
className[10][7][1]= "ACE_Remote_Name_Space";// [OS]
className[11][1][1]= "ACE_TSS_Ref";
className[11][1][2]= "ACE_TSS_Info";
className[11][1][3]= "ACE_TSS_Cleanup";
className[11][1][4]= "ACE_Thread_Adapter";
className[11][1][5]= "ACE_OS";// [Reactor]
className[12][1][1]= "ACE_Event_Handler ";
```

```
className[12][2][1]= "ACE_Event_Handler_T";
className[12][3][1]= "ACE_Handle_Set";
className[12][3][2]= "ACE_Handle_Set_Iterator";
className[12][4][1]= "ACE_Overlapped_IO";
className[12][4][2]= "ACE_Proactor";
className[12][4][3]= "ACE_Overlapped_File";
className[12][4][4]= "ACE_Reactor_Notify";
className[12][4][5]= "ACE_Reactor_Token";
className[12][4][6]= "ACE_Handler_Repository";
className[12][4][7]= "ACE_Handler_Repository_Iterator";
className[12][4][8]= "ACE_Reactor";
className[12][5][1]= "ACE_ReactorEx_Token";
className[12][5][2]= "ACE_ReactorEx_Notify";
className[12][5][3]= "ACE_ReactorEx";
className[12][6][1]= "ACE_Sig_Handlers_Set";
className[12][6][2]= "ACE_Sig_Set";
className[12][6][3]= "ACE_Sig_Action";
className[12][6][4]= "ACE_Sig_Guard";
className[12][6][5]= "ACE_Sig_Handler";
className[12][6][6]= "ACE_Sig_Adapter";
className[12][6][7]= "ACE_Sig_Handlers";
className[12][7][1]= "ACE_Time_Value";
className[12][7][2]= "ACE_Countdown_Time";
className[12][8][1]= "ACE_Timer_Queue";// [System_V_IPC]
className[13][1][1]= "ACE_SV_Message";
className[13][2][1]= "ACE_SV_Message_Queue";
className[13][3][1]= "ACE_Typed_SV_Message";
className[13][4][1]= "ACE_Typed_SV_Message_Queue";
className[13][5][1]= "ACE_SV_Semaphore_Complex";
className[13][6][1]= "ACE_SV_Semaphore_Simple";
className[13][7][1]= "ACE_SV_Shared_Memory";// [Timers]
className[14][1][1]= "ACE_Profile_Timer";
className[14][1][2]= "ACE_Profile_Timer";
className[14][2][1]= "ACE_High_Res_Timer";// [Token_Services]
className[15][1][1]= "ACE_TOKEN_CONST";
className[15][1][2]= "ACE_TPQ_Entry";
className[15][1][3]= "ACE_TSS_TPQ_Entry";
className[15][1][4]= "ACE_TPQ_Iterator";
className[15][1][5]= "ACE_Token_Proxy_Queue";
className[15][1][6]= "ACE_Tokens";
className[15][1][7]= "ACE_Local_Mutex";
className[15][1][8]= "ACE_Mutex_Token";
className[15][1][9]= "ACE_RW_Token";
className[15][1][10]= "ACE_Token_Name";
className[15][1][11]= "ACE_Token_Proxy";
className[15][1][12]= "ACE_Null_Token";
className[15][1][13]= "ACE_Local_Mutex";
className[15][1][14]= "ACE_Local_RLock";
className[15][1][15]= "ACE_Local_WLock";
className[15][2][1]= "ACE_Token_Name";
className[15][3][1]= "ACE_Remote_Token_Proxy";
className[15][3][2]= "ACE_Remote_Mutex";
className[15][3][3]= "ACE_Remote_RLock";
className[15][3][4]= "ACE_Remote_WLock";
className[15][3][5]= "ACE_TSS_Connection";
className[15][4][1]= "ACE_Token_Collection";
className[15][5][1]= "ACE_Token_Manager";
className[15][6][1]= "ACE_Mutex_Invariants";
```

```
  className[15][6][2]= "ACE_RWLock_Invariants";
  className[15][6][3]= "ACE_Token_Invariant_Manager";
  className[15][7][1]= "ACE_Token_Request";
  className[15][7][2]= "ACE_Token_Reply";

// Following I will zero all MPs, MRs, etc. since that it is not easy
// to extract all these details from ACE.
// This example on ACE will check for inter-class inheritance relation

//number of public methods per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
        MP[i][j][k]=0;
      }
    }
  }

//number of protected methods per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
        MR[i][j][k]=0;
      }
    }
  }

//number of private methods per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
        MV[i][j][k]=0;
      }
    }
  }

//number of public objects per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
        OP[i][j][k]=0;
      }
    }
  }

//number of protected methods per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
        OR[i][j][k]=0;
      }
    }
  }

//number of private methods per class per component per framework
  for (int i=1; i<=F; i++){
    for (int j=1; j<=C[i]; j++){
      for (int k=1; k<=c[i][j];k++){
```

```
          OV[i][j][k]=0;
        }
      }
    }

 //number of classes inherited in public per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhP[i][j][k]=0;
       }
     }
   }//number of classes inherited in protected per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhR[i][j][k]=0;
       }
     }
   }

 //number of classes inherited in private per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhV[i][j][k]=0;
       }
     }
   }//number of classes inherited from shared pools in public per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhPS[i][j][k]=0;
       }
     }
   }

 //number of classes inherited from shared pools in protected per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhRS[i][j][k]=0;
       }
     }
   }
 //number of classes inherited  from shared pools in private per class per component per framework
   for (int i=1; i<=F; i++){
     for (int j=1; j<=C[i]; j++){
       for (int k=1; k<=c[i][j];k++){
         MinhVS[i][j][k]=0;
       }
     }
   }//*********************************************************
 // STEP#2: instantiate for the Frameworks
 // objects and the embedied Component, MainSection,
 // Class, PublicSection, ProtectedSection, PrivateSection,
 // Method, etc. etc, objects
 // ALSO: Assign ID values for each object
```

```
// *****************************************************
// instantiate the framework objects
  for (int i=1; i<=F; i++){
    Frameworks[i]=new FrameworkNode();
    Frameworks[i].FrameworkNodeNumber=i;

    //instantiate the component objects for this new framework object
    for (int j=1; j<=C[i]; j++){
      Frameworks[i].Component[j]=new ComponentNode();
      Frameworks[i].Component[j].ID.ComponentNodeNumber=j;
      Frameworks[i].Component[j].ID.ParentFrameworkNodeNumber=i;
      Frameworks[i].Component[j].MainSection="main_"+i+"_"+j;
      //instantiate the class objects for this component object
      for (int k=1; k <= c[i][j]; k++){
        Frameworks[i].Component[j].Class[k]=new ClassNode();
        Frameworks[i].Component[j].Class[k].ID.ClassNodeNumber=k;
        Frameworks[i].Component[j].Class[k].ID.ClassName=className[i][j][k];
        Frameworks[i].Component[j].Class[k].ID.ParentComponent.ComponentNodeNumber=j;
        Frameworks[i].Component[j].Class[k].ID.ParentComponent.ParentFrameworkNodeNumber=i;
        //instantiate for the public section of this class
        Frameworks[i].Component[j].Class[k].PublicSection=new PublicSectionNode();
        for (int l=1; l<=MP[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].PublicSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].PublicSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
        //instantiate for the protected section of this class
        Frameworks[i].Component[j].Class[k].ProtectedSection=new ProtectedSectionNode();
        for (int l=1; l<=MR[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].ProtectedSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
        //instantiate for the private section of this class
        Frameworks[i].Component[j].Class[k].PrivateSection=new PrivateSectionNode();
        for (int l=1; l<=MV[i][j][k];l++){
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l]=new MethodNode();
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ReturnedValueType="void";
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.MethodNodeNumber=l;
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ClassNodeNum-
ber=k;
          Frameworks[i].Component[j].Class[k].PrivateSection.Method[l].ID.ParentClass.ParentCompo-
nent.ComponentNodeNumber=j;
          Frameworks[i].Component[j].Class[k].PrivateSection.Me-
thod[l].ID.ParentClass.ParentComponent.ParentFrameworkNodeNumber=i;
        }
```

```
      }
    }
  }

//*******************************************************************
// STEP#4: read in the inheritance references (CLassIDs) for each cless
//*******************************************************************
// [ASX] NONE// [CORBA] //class ACE_ST_CORBA_Handler
—> [2][1][1]
  Frameworks[2].Component[1].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[2].Component[1].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[2].Component[1].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
  Frameworks[2].Component[1].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=2;
  MinhP[2][1][2]=1; //class ACE_MT_CORBA_Handler
—> [2][1][1]
  Frameworks[2].Component[1].Class[3].PublicInheritedClasses[1]= new ClassID();
  Frameworks[2].Component[1].Class[3].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[2].Component[1].Class[3].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
  Frameworks[2].Component[1].Class[3].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=2;
  MinhP[2][1][3]=1;
// [Collections]  NONE// [Concurrency]
  //class ACE_RW_Process_Mutex   —> [4][4][6]
  Frameworks[4].Component[4].Class[7].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[7].PublicInheritedClasses[1].ClassNodeNumber=6;
  Frameworks[4].Component[4].Class[7].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=4;
  Frameworks[4].Component[4].Class[7].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][7]=1; //class ACE_Manual_Event   —> [4][4][12]
  Frameworks[4].Component[4].Class[13].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[13].PublicInheritedClasses[1].ClassNodeNumber=12;
  Frameworks[4].Component[4].Class[13].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[13].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][13]=1;  //class ACE_Auto_Event    —> [4][4][12]
  Frameworks[4].Component[4].Class[14].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[14].PublicInheritedClasses[1].ClassNodeNumber=12;
  Frameworks[4].Component[4].Class[14].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[14].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][14]=1;  //class ACE_RW_Thread_Mutex
—> [4][4][4]
  Frameworks[4].Component[4].Class[19].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[19].PublicInheritedClasses[1].ClassNodeNumber=4;
  Frameworks[4].Component[4].Class[19].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[19].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][19]=1;  //class ACE_Thread_Semaphore
—> [4][4][2]
  Frameworks[4].Component[4].Class[20].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[20].PublicInheritedClasses[1].ClassNodeNumber=2;
```

```
  Frameworks[4].Component[4].Class[20].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[20].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][20]=1; //class ACE_Process_Barrier ---> [4][4][21]
  Frameworks[4].Component[4].Class[22].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[22].PublicInheritedClasses[1].ClassNodeNumber=21;
  Frameworks[4].Component[4].Class[22].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[22].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][22]=1; //class ACE_Thread_Barrier ---> [4][4][21]
  Frameworks[4].Component[4].Class[23].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[4].Class[23].PublicInheritedClasses[1].ClassNodeNumber=21;
  Frameworks[4].Component[4].Class[23].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=4;
  Frameworks[4].Component[4].Class[23].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=4;
  MinhP[4][4][23]=1; //class ACE_Test_and_Set    ---> [12][1][1]
  Frameworks[4].Component[6].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[4].Component[6].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[4].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
  Frameworks[4].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[4][6][1]=1;// [Connection]
  //class ACE_Strategy_Acceptor     ---> [5][1][1]
  Frameworks[5].Component[1].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[5].Component[1].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[5].Component[1].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
  Frameworks[5].Component[1].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=5;
  MinhP[5][1][2]=1; //class ACE_Reactor_Notification_Strategy
---> [5][4][3]
  Frameworks[5].Component[4].Class[4].PublicInheritedClasses[1]= new ClassID();
  Frameworks[5].Component[4].Class[4].PublicInheritedClasses[1].ClassNodeNumber=3;
  Frameworks[5].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=4;
  Frameworks[5].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=5;
  MinhP[5][4][4]=1;

  //class ACE_ReactorEx_Notification_Strategy ---> [5][4][3]
  Frameworks[5].Component[4].Class[5].PublicInheritedClasses[1]= new ClassID();
  Frameworks[5].Component[4].Class[5].PublicInheritedClasses[1].ClassNodeNumber=3;
  Frameworks[5].Component[4].Class[5].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=4;
  Frameworks[5].Component[4].Class[5].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=5;
  MinhP[5][4][5]=1;

// [IPC]
  //class ACE_DEV     ---> [6][1][1]
  Frameworks[6].Component[2].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
```

```
 Frameworks[6].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][2][1]=1;

 //class ACE_DEV_Connector     ---> [6][2][1]
 Frameworks[6].Component[3].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[3].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=2;
 Frameworks[6].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][3][1]=1;

 //class ACE_DEV_IO    ---> [6][2][1]
 Frameworks[6].Component[4].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[4].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[4].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=2;
 Frameworks[6].Component[4].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][4][1]=1; //class ACE_FILE     ---> [6][1][1]
 Frameworks[6].Component[5].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[5].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[5].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=1;
 Frameworks[6].Component[5].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][5][1]=1; //class ACE_FILE_Connector  ---> [6][5][1]
 Frameworks[6].Component[6].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[6].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=5;
 Frameworks[6].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][6][1]=1; //class ACE_FILE_IO   ---> [6][5][1]
 Frameworks[6].Component[7].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[7].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=5;
 Frameworks[6].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][7][1]=1; //class ACE_DEV_Addr     ---> [6][9][1]
 Frameworks[6].Component[10].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[10].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[10].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=10;
 Frameworks[6].Component[10].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][10][1]=1; //class ACE_FILE_Addr    ---> [6][9][1]
 Frameworks[6].Component[11].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[11].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
 Frameworks[6].Component[11].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=11;
 Frameworks[6].Component[11].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
 MinhP[6][11][1]=1; //class ACE_INET_Addr    ---> [6][9][1]
 Frameworks[6].Component[12].Class[1].PublicInheritedClasses[1]= new ClassID();
 Frameworks[6].Component[12].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
```

```
  Frameworks[6].Component[12].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=9;
  Frameworks[6].Component[12].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][12][1]=1; //class ACE_SPIPE_Addr ——> [6][9][1]
  Frameworks[6].Component[13].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[13].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[13].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=9;
  Frameworks[6].Component[13].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][13][1]=1; //class ACE_UNIX_Addr ——> [6][9][1]
  Frameworks[6].Component[14].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[14].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[14].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=9;
  Frameworks[6].Component[14].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][14][1]=1; //class ACE_FIFO ——> [6][8][1]
  Frameworks[6].Component[15].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[15].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[15].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=15;
  Frameworks[6].Component[15].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][15][1]=1; //class ACE_FIFO_Recv ——> [6][15][1]
  Frameworks[6].Component[16].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[16].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[16].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=15;
  Frameworks[6].Component[16].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][16][1]=1; //class ACE_FIFO_Recv_Msg
——> [6][16][1]
  Frameworks[6].Component[17].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[17].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[17].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=16;
  Frameworks[6].Component[17].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][17][1]=1; //class ACE_FIFO_Send ——> [6][15][1]
  Frameworks[6].Component[18].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[18].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[18].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=15;
  Frameworks[6].Component[18].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][18][1]=1; //class ACE_FIFO_Send_Msg
——> [6][18][1]
  Frameworks[6].Component[19].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[19].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[19].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=18;
  Frameworks[6].Component[19].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][19][1]=1; //class ACE_LSOCK_Acceptor
——> [6][26][1]
  Frameworks[6].Component[21].Class[1].PublicInheritedClasses[1]= new ClassID();
```

Frameworks[6].Component[21].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[21].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-Number=26;
Frameworks[6].Component[21].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-deNumber=6;
MinhP[6][21][1]=1; //class ACE_LSOCK_CODgram
—> [6][27][1], [6][20][1]
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-Number=27;
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-deNumber=6;
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[2]= new ClassID();
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[2].ClassNodeNumber=1;
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[2].ParentComponent.ComponentNode-Number=20;
Frameworks[6].Component[22].Class[1].PublicInheritedClasses[2].ParentComponent.ParentFrameworkNo-deNumber=6;
MinhP[6][22][1]=2; //class ACE_LSOCK_Connector
—> [6][28][1]
Frameworks[6].Component[23].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[23].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[23].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-Number=28;
Frameworks[6].Component[23].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-deNumber=6;
MinhP[6][23][1]=1; //6–24–1 class ACE_LSOCK_Dgram
—> [6][29][1], [6][20][1]
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-Number=29;
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-deNumber=6;
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[2]= new ClassID();
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[2].ClassNodeNumber=1;
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[2].ParentComponent.ComponentNode-Number=20;
Frameworks[6].Component[24].Class[1].PublicInheritedClasses[2].ParentComponent.ParentFrameworkNo-deNumber=6;
MinhP[6][24][1]=2; //6–24–2 class ACE_LSOCK_Stream
—> [6][33][1], [6][20][1]
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNode-Number=33;
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-deNumber=6;
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[2]= new ClassID();
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[2].ClassNodeNumber=1;
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[2].ParentComponent.ComponentNode-Number=20;
Frameworks[6].Component[24].Class[2].PublicInheritedClasses[2].ParentComponent.ParentFrameworkNo-deNumber=6;
MinhP[6][24][2]=2; //6–25–1 class ACE_SOCK
—> [6][8][1]
Frameworks[6].Component[25].Class[1].PublicInheritedClasses[1]= new ClassID();

Frameworks[6].Component[25].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[25].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=8;
Frameworks[6].Component[25].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][25][1]=1; //6–26–1 class ACE_SOCK_Acceptor
—> [6][25][1]
Frameworks[6].Component[26].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[26].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[26].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=25;
Frameworks[6].Component[26].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][26][1]=1; //6–27–1 class ACE_SOCK_CODgram
—> [6][32][1]
Frameworks[6].Component[27].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[27].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[27].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=32;
Frameworks[6].Component[27].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][27][1]=1; //6–28–1 class ACE_SOCK_Connector
—> [6][25][1]
Frameworks[6].Component[28].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[28].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[28].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=25;
Frameworks[6].Component[28].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][28][1]=1; //6–29–1 class ACE_SOCK_Dgram
—> [6][25][1]
Frameworks[6].Component[29].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[29].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[29].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=25;
Frameworks[6].Component[29].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][29][1]=1; //6–30–1 class ACE_SOCK_Dgram_Bcast
—> [6][29][1]
Frameworks[6].Component[30].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[30].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[30].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=29;
Frameworks[6].Component[30].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][30][1]=1; //6–31–1 class ACE_SOCK_Dgram_Mcast
—> [6][29][1]
Frameworks[6].Component[31].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[31].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[31].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=29;
Frameworks[6].Component[31].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][31][1]=1; //6–32–1 class ACE_SOCK_IO
—> [6][25][1]
Frameworks[6].Component[32].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[32].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[32].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-

```
Number=25;
  Frameworks[6].Component[32].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][32][1]=1; //6–33–1  class ACE_SOCK_Stream
—–> [6][32][1]
  Frameworks[6].Component[33].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[33].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[33].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=32;
  Frameworks[6].Component[33].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][33][1]=1; //6–34–1  class ACE_SPIPE
—–> [6][8][1]
  Frameworks[6].Component[34].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[34].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[34].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=8;
  Frameworks[6].Component[34].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][34][1]=1; //6–35–1  class ACE_SPIPE_Acceptor
—–> [6][34][1]
  Frameworks[6].Component[35].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[35].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[35].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=34;
  Frameworks[6].Component[35].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][35][1]=1; //6–36–1  class ACE_SPIPE_Connector
—–> [6][34][1]
  Frameworks[6].Component[36].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[36].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[36].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=34;
  Frameworks[6].Component[36].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][36][1]=1; //6–37–1  class ACE_SPIPE_Stream
—–> [6][34][1]
  Frameworks[6].Component[37].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[37].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[37].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=34;
  Frameworks[6].Component[37].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][37][1]=1; //6–38–1  class ACE_TLI      —–> [6][8][1]
  Frameworks[6].Component[38].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[38].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[38].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=8;
  Frameworks[6].Component[38].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
  MinhP[6][38][1]=1; //6–39–3  class ACE_TLI_Acceptor
—–> [6][38][1]
  Frameworks[6].Component[39].Class[3].PublicInheritedClasses[1]= new ClassID();
  Frameworks[6].Component[39].Class[3].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[6].Component[39].Class[3].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=38;
  Frameworks[6].Component[39].Class[3].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=6;
```

MinhP[6][39][3]=1; //6–40–1 class ACE_TLI_Connector
—> [6][38][1]
Frameworks[6].Component[40].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[40].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[40].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=38;
Frameworks[6].Component[40].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][40][1]=1; //6–41–1 class ACE_TLI_Stream
—> [6][38][1]
Frameworks[6].Component[41].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[41].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[41].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=38;
Frameworks[6].Component[41].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][41][1]=1; //6–42–1 class ACE_UPIPE_Acceptor
—> [6][35][1]
Frameworks[6].Component[42].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[42].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[42].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=35;
Frameworks[6].Component[42].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][42][1]=1; //6–43–1 class ACE_UPIPE_Connector
—> [6][34][1]
Frameworks[6].Component[43].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[43].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[43].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=34;
Frameworks[6].Component[43].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][43][1]=1; //6–44–1 class ACE_UPIPE_Stream
—> [6][34][1]
Frameworks[6].Component[44].Class[1].PublicInheritedClasses[1]= new ClassID();
Frameworks[6].Component[44].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[6].Component[44].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=34;
Frameworks[6].Component[44].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=6;
MinhP[6][44][1]=1;
// [Log_Msg] NONE// [Memory]
//8–4–4 class ACE_Shared_Memory_Pool —> [12][1][1]
Frameworks[8].Component[4].Class[4].PublicInheritedClasses[1]= new ClassID();
Frameworks[8].Component[4].Class[4].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[8].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=1;
Frameworks[8].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=12;
MinhP[8][4][4]=1; //8–4–8 class ACE_MMAP_Memory_Pool
—> [12][1][1]
Frameworks[8].Component[4].Class[8].PublicInheritedClasses[1]= new ClassID();
Frameworks[8].Component[4].Class[8].PublicInheritedClasses[1].ClassNodeNumber=1;
Frameworks[8].Component[4].Class[8].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=1;
Frameworks[8].Component[4].Class[8].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=12;
MinhP[8][4][8]=1; //8–4–9 class ACE_Lite_MMAP_Memory_Pool —> [8][4][9]

```
  Frameworks[8].Component[4].Class[9].PublicInheritedClasses[1]= new ClassID();
  Frameworks[8].Component[4].Class[9].PublicInheritedClasses[1].ClassNodeNumber=9;
  Frameworks[8].Component[4].Class[9].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=4;
  Frameworks[8].Component[4].Class[9].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=8;
  MinhP[8][4][9]=1; //8–6–1 class ACE_Shared_Memory_MM
—> [8][5][1]
  Frameworks[8].Component[6].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[8].Component[6].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[8].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=5;
  Frameworks[8].Component[6].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=8;
  MinhP[8][6][1]=1; //8–7–1 class ACE_Shared_Memory_SV
—> [8][5][1]
  Frameworks[8].Component[7].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[8].Component[7].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[8].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNum-
ber=5;
  Frameworks[8].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=8;
  MinhP[8][7][1]=1;
// [Misc] NONE// [Name_Service]
  //10–2–2 class ACE_Local_Name_Space —> [10][5][2]
  Frameworks[10].Component[2].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[10].Component[2].Class[2].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[10].Component[2].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=5;
  Frameworks[10].Component[2].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=10;
  MinhP[10][2][2]=1; //10–3–1 class ACE_Name_Proxy
—> [12][1][1]
  Frameworks[10].Component[3].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[10].Component[3].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[10].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[10].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[10][3][1]=1; //10–7–1 class ACE_Remote_Name_Space
—> [10][5][2]
  Frameworks[10].Component[7].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[10].Component[7].Class[1].PublicInheritedClasses[1].ClassNodeNumber=2;
  Frameworks[10].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=5;
  Frameworks[10].Component[7].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=10;
  MinhP[10][7][1]=1;
// [OS] NONE// [Reactor]
  //12–2–1 class ACE_Event_Handler_T —> [12][1][1]
  Frameworks[12].Component[2].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[2].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[12].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[12].Component[2].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][2][1]=1; //12–4–2 class ACE_Proactor
—> [12][1][1]
```

```
  Frameworks[12].Component[4].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[4].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[12].Component[4].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[12].Component[4].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][4][2]=1; //12-4-4  class ACE_Reactor_Notify
—> [12][1][1]
  Frameworks[12].Component[4].Class[4].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[4].Class[4].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[12].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[12].Component[4].Class[4].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][4][4]=1; //12-5-2  class ACE_ReactorEx_Notify
—> [12][1][1]
  Frameworks[12].Component[5].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[5].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[12].Component[5].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[12].Component[5].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][5][2]=1; //12-6-6  class ACE_Sig_Adapter
—> [12][1][1]
  Frameworks[12].Component[6].Class[6].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[6].Class[6].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[12].Component[6].Class[6].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[12].Component[6].Class[6].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][6][6]=1; //12-6-7  class ACE_Sig_Handlers
—> [12][6][5]
  Frameworks[12].Component[6].Class[7].PublicInheritedClasses[1]= new ClassID();
  Frameworks[12].Component[6].Class[7].PublicInheritedClasses[1].ClassNodeNumber=5;
  Frameworks[12].Component[6].Class[7].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=6;
  Frameworks[12].Component[6].Class[7].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=12;
  MinhP[12][6][7]=1;
// [System_V_IPC]
  //13-5-1  class ACE_SV_Semaphore_Complex  —> private [13][6][1]
  Frameworks[13].Component[5].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[13].Component[5].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[13].Component[5].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=6;
  Frameworks[13].Component[5].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=13;
  MinhP[13][5][1]=1;
// [Timers]  NONE// [Token_Services]
  //15-1-8  class ACE_Mutex_Token    —> [15][1][6]
  Frameworks[15].Component[1].Class[8].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[8].PublicInheritedClasses[1].ClassNodeNumber=6;
  Frameworks[15].Component[1].Class[8].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[8].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][1][8]=1; //15-1-9  class ACE_RW_Token
—> [15][1][6]
```

```
  Frameworks[15].Component[1].Class[9].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[9].PublicInheritedClasses[1].ClassNodeNumber=6;
  Frameworks[15].Component[1].Class[9].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[9].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][1][9]=1; //15-1-12 class ACE_Null_Token
—> [15][1][11]
  Frameworks[15].Component[1].Class[12].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[12].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[1].Class[12].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[12].PublicInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=15;
  MinhP[15][1][12]=1; //15-1-13 class ACE_Local_Mutex
—> [15][1][11]
  Frameworks[15].Component[1].Class[13].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[13].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[1].Class[13].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[13].PublicInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=15;
  MinhP[15][1][13]=1; //15-1-14 class ACE_Local_RLock
—> [15][1][11]
  Frameworks[15].Component[1].Class[14].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[14].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[1].Class[14].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[14].PublicInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=15;
  MinhP[15][1][14]=1; //15-1-15 class ACE_Local_WLock
—> [15][1][11]
  Frameworks[15].Component[1].Class[15].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[1].Class[15].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[1].Class[15].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[1].Class[15].PublicInheritedClasses[1].ParentComponent.ParentFramework-
NodeNumber=15;
  MinhP[15][1][15]=1; //15-3-1  class ACE_Remote_Token_Proxy
—> [15][1][11]
  Frameworks[15].Component[3].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[3].Class[1].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[3].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][3][1]=1; //15-3-2  class ACE_Remote_Mutex
—> [15][3][1]
  Frameworks[15].Component[3].Class[2].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[3].Class[2].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[15].Component[3].Class[2].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=3;
  Frameworks[15].Component[3].Class[2].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][3][2]=1; //15-3-3  class ACE_Remote_RLock
—> [15][3][1]
  Frameworks[15].Component[3].Class[3].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[3].Class[3].PublicInheritedClasses[1].ClassNodeNumber=1;
```

```
  Frameworks[15].Component[3].Class[3].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=3;
  Frameworks[15].Component[3].Class[3].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][3][3]=1; //15-3-4  class ACE_Remote_WLock
—> [15][3][1]
  Frameworks[15].Component[3].Class[4].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[3].Class[4].PublicInheritedClasses[1].ClassNodeNumber=1;
  Frameworks[15].Component[3].Class[4].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=3;
  Frameworks[15].Component[3].Class[4].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][3][4]=1; //15-4-1  class ACE_Token_Collection
—> [15][1][11]
  Frameworks[15].Component[4].Class[1].PublicInheritedClasses[1]= new ClassID();
  Frameworks[15].Component[4].Class[1].PublicInheritedClasses[1].ClassNodeNumber=11;
  Frameworks[15].Component[4].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNode-
Number=1;
  Frameworks[15].Component[4].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNo-
deNumber=15;
  MinhP[15][4][1]=1;
}
protected static void initializeSharedPools_Example_1() {
// ****************************************
// STEP#1 read values for:
//      F, C[], c[][],
//      MP[][][], MR[][][], MV[][][]
//      OP[][][], OR[][][], OV[][][]
// ****************************************
//number of shared pools
  S=2;
//number of shared classes per shared pool
  SC[1]=2;
  SC[2]=2;
//number of public methods per class per shared pool
  SMP[1][1]=2;SMP[1][2]=2;
  SMP[2][1]=2;SMP[2][2]=2;
//number of protected methods per class per shared pool
  SMR[1][1]=1;  SMR[1][2]=1;
  SMR[2][1]=1;  SMR[2][2]=1;
//number of private methods per class per shared pool
  SMV[1][1]=1;  SMV[1][2]=1;
  SMV[2][1]=1;  SMV[2][2]=1;
//number of public objects per class per shared pool
  SOP[1][1]=0; SOP[1][2]=1;
  SOP[2][1]=1; SOP[2][2]=0;
//number of protected methods per class  per shared pool
  SOR[1][1]=2;SOR[1][2]=2;
  SOR[2][1]=2;SOR[2][2]=2;
//number of private methods per class per shared pool
  SOV[1][1]=4;SOV[1][2]=4;
  SOV[2][1]=4;SOV[2][2]=4;
//number of classes inherited in public per class per shared pool
  SMinhP[1][1]=1; SMinhP[1][2]=0;
  SMinhP[2][1]=1; SMinhP[2][2]=0;
//number of classes inherited in protected per class per shared pool
  SMinhR[1][1]=0;SMinhR[1][2]=0;
  SMinhR[2][1]=0;SMinhR[2][2]=0;
```

```
//number of classes inherited in private per class per shared pool
  SMinhV[1][1]=0;SMinhV[1][2]=0;
  SMinhV[2][1]=0;SMinhV[2][2]=0;//number of classes inherited in public from frameworks per class per
shared pool
  SMinhPF[1][1]=0; SMinhPF[1][2]=0;
  SMinhPF[2][1]=0; SMinhPF[2][2]=0;
//number of classes inherited in protected from frameworks per class per shared pool
  SMinhRF[1][1]=0; SMinhRF[1][2]=0;
  SMinhRF[2][1]=0; SMinhRF[2][2]=0;
//number of classes inherited in private from frameworks per class per shared pool
  SMinhVF[1][1]=0; SMinhVF[1][2]=0;
  SMinhVF[2][1]=0; SMinhVF[2][2]=0;//
*****************************************************
// STEP#2: instantiate for the Frameworks
// objects and the embedied
// Class, PublicSection, ProtectedSection, PrivateSection,
// Method, etc. etc, objects
// ALSO: Assign ID values for each object
//*****************************************************
// instantiate the framework objects
  for (int i=1; i<=S; i++){
    SharedPools[i]=new SharedPoolNode();
    SharedPools[i].SharedPoolNodeNumber=i;
    //instantiate the shared classes objects for this new shared pool object
    for (int j=1; j<=SC[i]; j++){
      SharedPools[i].Class[j]=new SharedClassNode();
      SharedPools[i].Class[j].ClassName="s_class"+i+j;
      SharedPools[i].Class[j].ID.ClassNodeNumber=j;
      SharedPools[i].Class[j].ID.ParentSharedPoolNodeNumber=i;
      //instantiate for the public section of this class
      SharedPools[i].Class[j].PublicSection=new SharedPublicSectionNode();
      for (int k=1; k<=SMP[i][j];k++){
        SharedPools[i].Class[j].PublicSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PublicSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PublicSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
      }
      //instantiate for the protected section of this class
      SharedPools[i].Class[j].ProtectedSection=new SharedProtectedSectionNode();
      for (int k=1; k<=SMR[i][j];k++){
        SharedPools[i].Class[j].ProtectedSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].ProtectedSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNode-
Number=i;
      }
      //instantiate for the private section of this class
      SharedPools[i].Class[j].PrivateSection=new SharedPrivateSectionNode();
      for (int k=1; k<=SMV[i][j];k++){
        SharedPools[i].Class[j].PrivateSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PrivateSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
```

```
      }
    }
  }

// ****************************************************
// STEP#3:  Read the entries for each of these objects
// ****************************************************//SharedCLASS(1,1)
    SharedPools[1].Class[1].PublicSection.Method[1].MethodName="s_p_mthd111";
    SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
    SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="class112";
    SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Y";
    SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="s_class12";
    SharedPools[1].Class[1].PublicSection.Method[1].ReturnedValueType="T5";
  SharedPools[1].Class[1].PublicSection.Method[2].MethodName="s_p_mthd112";
    SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
    SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
    SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
    SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
    SharedPools[1].Class[1].PublicSection.Method[2].ReturnedValueType="T7";
  SharedPools[1].Class[1].ProtectedSection.Method[1].MethodName="s_r_mthd111";
    SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
    SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T7";
    SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgName[1]="Y";
    SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgType[1]="T2";

    SharedPools[1].Class[1].ProtectedSection.Object[1]="s_r_obj112";
    SharedPools[1].Class[1].ProtectedSection.ObjectType[1]="class112";
    SharedPools[1].Class[1].ProtectedSection.Object[2]="s_r_obj112";
    SharedPools[1].Class[1].ProtectedSection.ObjectType[2]="T2";

    SharedPools[1].Class[1].PrivateSection.Method[1].MethodName="s_v_mthd111";
    SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgName[1]="W";
    SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T6";
    SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgName[1]="G";
    SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgType[1]="s_class22";
  SharedPools[1].Class[1].PrivateSection.Object[1]="s_v_obj111";
    SharedPools[1].Class[1].PrivateSection.ObjectType[1]="s_class21";
    SharedPools[1].Class[1].PrivateSection.Object[2]="s_v_obj112";
    SharedPools[1].Class[1].PrivateSection.ObjectType[2]="class221";
    SharedPools[1].Class[1].PrivateSection.Object[3]="s_v_objt113";
    SharedPools[1].Class[1].PrivateSection.ObjectType[3]="T9";
    SharedPools[1].Class[1].PrivateSection.Object[4]="s_v_obj114";
    SharedPools[1].Class[1].PrivateSection.ObjectType[4]="T8";
  //SharedCLASS(1,2)
    SharedPools[1].Class[2].PublicSection.Method[1].MethodName="s_p_mthd1121";
    SharedPools[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="A";
    SharedPools[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
    SharedPools[1].Class[2].PublicSection.Method[1].ReturnedValueType="T4";
  SharedPools[1].Class[2].PublicSection.Method[2].MethodName="s_p_mthd122";
    SharedPools[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
    SharedPools[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="T5";
    SharedPools[1].Class[2].PublicSection.Method[2].ReturnedValueType="T4";
  SharedPools[1].Class[2].PublicSection.Object[1]="s_p_obj121";
    SharedPools[1].Class[2].PublicSection.ObjectType[1]="T5";
  SharedPools[1].Class[2].ProtectedSection.Method[1].MethodName="s_r_mthd121";
    SharedPools[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
  SharedPools[1].Class[2].ProtectedSection.Object[1]="s_r_obj121";
    SharedPools[1].Class[2].ProtectedSection.ObjectType[1]="s_class12";
```

```
SharedPools[1].Class[2].ProtectedSection.Object[2]="s_r_obj122";
SharedPools[1].Class[2].ProtectedSection.ObjectType[2]="T5";
SharedPools[1].Class[2].PrivateSection.Method[1].MethodName="s_v_mthd121";
SharedPools[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T3";
SharedPools[1].Class[2].PrivateSection.Object[1]="s_v_obj121";
SharedPools[1].Class[2].PrivateSection.ObjectType[1]="s_class11";
SharedPools[1].Class[2].PrivateSection.Object[2]="s_v_obj122";
SharedPools[1].Class[2].PrivateSection.ObjectType[2]="T6";
SharedPools[1].Class[2].PrivateSection.Object[3]="s_v_obj123";
SharedPools[1].Class[2].PrivateSection.ObjectType[3]="T7";
SharedPools[1].Class[2].PrivateSection.Object[4]="s_v_obj124";
SharedPools[1].Class[2].PrivateSection.ObjectType[4]="T1";

//SharedCLASS(2,1)
SharedPools[2].Class[1].PublicSection.Method[1].MethodName="s_p_mthd211";
SharedPools[2].Class[1].PublicSection.Method[1].ReturnedValueType="T8";

SharedPools[2].Class[1].PublicSection.Method[2].MethodName="s_p_mthd212";
SharedPools[2].Class[1].PublicSection.Method[2].ReturnedValueType="T5";
SharedPools[2].Class[1].PublicSection.Object[1]="s_p_obj211";
SharedPools[2].Class[1].PublicSection.ObjectType[1]="T12";
SharedPools[2].Class[1].ProtectedSection.Method[1].MethodName="s_r_mthd211";
SharedPools[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T1";
SharedPools[2].Class[1].ProtectedSection.Object[1]="s_r_obj211";
SharedPools[2].Class[1].ProtectedSection.ObjectType[1]="s_class12";
SharedPools[2].Class[1].ProtectedSection.Object[2]="s_r_obj212";
SharedPools[2].Class[1].ProtectedSection.ObjectType[2]="T13";
SharedPools[2].Class[1].PrivateSection.Method[1].MethodName="s_v_mthd211";
SharedPools[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
SharedPools[2].Class[1].PrivateSection.Object[1]="s_v_obj211";
SharedPools[2].Class[1].PrivateSection.ObjectType[1]="s_class21";
SharedPools[2].Class[1].PrivateSection.Object[2]="s_v_obj212";
SharedPools[2].Class[1].PrivateSection.ObjectType[2]="class22";
SharedPools[2].Class[1].PrivateSection.Object[3]="s_v_obj213";
SharedPools[2].Class[1].PrivateSection.ObjectType[3]="T7";
SharedPools[2].Class[1].PrivateSection.Object[4]="s_v_obj214";
SharedPools[2].Class[1].PrivateSection.ObjectType[4]="T6";
//SharedCLASS(2,2)
SharedPools[2].Class[2].PublicSection.Method[1].MethodName="s_p_mthd221";
SharedPools[2].Class[2].PublicSection.Method[2].MethodName="s_p_mthd222";
SharedPools[2].Class[2].PublicSection.Method[2].ReturnedValueType="T3";
SharedPools[2].Class[2].ProtectedSection.Method[1].MethodName="s_r_mthd221";
SharedPools[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T2";
SharedPools[2].Class[2].ProtectedSection.Object[1]="s_r_obj221";
SharedPools[2].Class[2].ProtectedSection.ObjectType[1]="T1";
SharedPools[2].Class[2].ProtectedSection.Object[2]="s_r_obj222";
SharedPools[2].Class[2].ProtectedSection.ObjectType[2]="T2";
SharedPools[2].Class[2].PrivateSection.Method[1].MethodName="s_v_mthd221";
SharedPools[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T7";
SharedPools[2].Class[2].PrivateSection.Object[1]="s_v_obj221";
SharedPools[2].Class[2].PrivateSection.ObjectType[1]="s_class12";
SharedPools[2].Class[2].PrivateSection.Object[2]="s_v_obj222";
SharedPools[2].Class[2].PrivateSection.ObjectType[2]="s_class21";
SharedPools[2].Class[2].PrivateSection.Object[3]="s_v_obj223";
SharedPools[2].Class[2].PrivateSection.ObjectType[3]="T6";
SharedPools[2].Class[2].PrivateSection.Object[4]="s_v_obj224";
SharedPools[2].Class[2].PrivateSection.ObjectType[4]="T9";//
*****************************************************************
```

```
// STEP#4: read in the inheritance references (CLassIDs) for each cless
//*********************************************************************
//SharedClass(1,1) inherits from two other classes in public —
  SharedPools[1].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
  SharedPools[1].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=2;
  SharedPools[1].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=1;//Class(2,1) inherits only from one other class in public

  SharedPools[2].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
  SharedPools[2].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=2;
  SharedPools[2].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;}
protected static void initializeSharedPools_Example_2() {
//****************************************
// STEP#1 read values for:
//      F, C[], c[][],
//      MP[][][], MR[][][], MV[][][]
//      OP[][][], OR[][][], OV[][][]
//****************************************
//number of shared pools
  S=2;
//number of shared classes per shared pool
  SC[1]=2;
  SC[2]=2;
//number of public methods per class per shared pool
  SMP[1][1]=2;SMP[1][2]=2;
  SMP[2][1]=2;SMP[2][2]=2;
//number of protected methods per class per shared pool
  SMR[1][1]=1;  SMR[1][2]=1;
  SMR[2][1]=1;  SMR[2][2]=1;
//number of private methods per class per shared pool
  SMV[1][1]=1;  SMV[1][2]=1;
  SMV[2][1]=1;  SMV[2][2]=1;
//number of public objects per class per shared pool
  SOP[1][1]=0; SOP[1][2]=1;
  SOP[2][1]=1; SOP[2][2]=0;
//number of protected methods per class  per shared pool
  SOR[1][1]=2;SOR[1][2]=2;
  SOR[2][1]=2;SOR[2][2]=2;
//number of private methods per class per shared pool
  SOV[1][1]=4;SOV[1][2]=4;
  SOV[2][1]=4;SOV[2][2]=4;
//number of classes inherited in public per class per shared pool
  SMinhP[1][1]=1; SMinhP[1][2]=0;
  SMinhP[2][1]=1; SMinhP[2][2]=0;
//number of classes inherited in protected per class per shared pool
  SMinhR[1][1]=0;SMinhR[1][2]=0;
  SMinhR[2][1]=0;SMinhR[2][2]=1;
//number of classes inherited in private per class per shared pool
  SMinhV[1][1]=1;SMinhV[1][2]=0;
  SMinhV[2][1]=0;SMinhV[2][2]=0;//number of classes inherited in public from frameworks per class per shared pool
  SMinhPF[1][1]=1; SMinhPF[1][2]=0;
  SMinhPF[2][1]=0; SMinhPF[2][2]=0;
//number of classes inherited in protected from frameworks per class per shared pool
  SMinhRF[1][1]=0; SMinhRF[1][2]=0;
  SMinhRF[2][1]=0; SMinhRF[2][2]=1;
//number of classes inherited in private from frameworks per class per shared pool
  SMinhVF[1][1]=0; SMinhVF[1][2]=0;
```

```
  SMinhVF[2][1]=0; SMinhVF[2][2]=0;//
***************************************************
// STEP#2: instantiate for the Frameworks
// objects and the embedied
// Class, PublicSection, ProtectedSection, PrivateSection,
// Method, etc. etc, objects
// ALSO: Assign ID values for each object
// ***************************************************// instantiate the framework objects
  for (int i=1; i<=S; i++){
    SharedPools[i]=new SharedPoolNode();
    SharedPools[i].SharedPoolNodeNumber=i;
    //instantiate the shared classes objects for this new shared pool object
    for (int j=1; j<=SC[i]; j++){
      SharedPools[i].Class[j]=new SharedClassNode();
      SharedPools[i].Class[j].ClassName="s_class"+i+j;
      SharedPools[i].Class[j].ID.ClassNodeNumber=j;
      SharedPools[i].Class[j].ID.ParentSharedPoolNodeNumber=i;
      //instantiate for the public section of this class
      SharedPools[i].Class[j].PublicSection=new SharedPublicSectionNode();
      for (int k=1; k<=SMP[i][j];k++){
        SharedPools[i].Class[j].PublicSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PublicSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PublicSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
      }
      //instantiate for the protected section of this class
      SharedPools[i].Class[j].ProtectedSection=new SharedProtectedSectionNode();
      for (int k=1; k<=SMR[i][j];k++){
        SharedPools[i].Class[j].ProtectedSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].ProtectedSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNode-
Number=i;
      }
      //instantiate for the private section of this class
      SharedPools[i].Class[j].PrivateSection=new SharedPrivateSectionNode();
      for (int k=1; k<=SMV[i][j];k++){
        SharedPools[i].Class[j].PrivateSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PrivateSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
      }
    }
  }

// ***************************************************
// STEP#3:  Read the entries for each of these objects
// ***************************************************//SharedCLASS(1,1)
//SharedClass(1,1) public section contains only two methods and no objects
//SMP[1][1]=2 && SOP[1][1]=0
  SharedPools[1].Class[1].PublicSection.Method[1].MethodName="s_p_mthd111";
  //get in ARGS for this method
  SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[1]="X";
```

```
SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[1]="class112";
SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgName[2]="Y";
SharedPools[1].Class[1].PublicSection.Method[1].ARGS.IN_ArgType[2]="s_class12";
SharedPools[1].Class[1].PublicSection.Method[1].ReturnedValueType="T5";
//————————
SharedPools[1].Class[1].PublicSection.Method[2].MethodName="s_p_mthd112";
//get in ARGS for this method
SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[1]="T1";
SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgName[2]="B";
SharedPools[1].Class[1].PublicSection.Method[2].ARGS.IN_ArgType[2]="T2";
SharedPools[1].Class[1].PublicSection.Method[2].ReturnedValueType="T7";//SharedClass(1,1) protected
section contains one method and 2 objects
//SMR[1][1]=1 && SOR[1][1]=2
SharedPools[1].Class[1].ProtectedSection.Method[1].MethodName="s_r_mthd111";
//get in ARGS for this method
SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgName[1]="X";
SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.IN_ArgType[1]="T7";
SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgName[1]="Y";
SharedPools[1].Class[1].ProtectedSection.Method[1].ARGS.OUT_ArgType[1]="T2";
//————————————————————
SharedPools[1].Class[1].ProtectedSection.Object[1]="s_r_obj112";
SharedPools[1].Class[1].ProtectedSection.ObjectType[1]="class112";
SharedPools[1].Class[1].ProtectedSection.Object[2]="s_r_obj112";
SharedPools[1].Class[1].ProtectedSection.ObjectType[2]="T2";

//SharedClass(1,1) private section contains 1 method and 4 objects
//SMV[1][1]=1 && SOV[1][1]=4
SharedPools[1].Class[1].PrivateSection.Method[1].MethodName="s_v_mthd111";
//ARGS for this method
SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgName[1]="W";
SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.IN_ArgType[1]="T6";
SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgName[1]="G";
SharedPools[1].Class[1].PrivateSection.Method[1].ARGS.OUT_ArgType[1]="s_class22";
//————————————————————
SharedPools[1].Class[1].PrivateSection.Object[1]="s_v_obj111";
SharedPools[1].Class[1].PrivateSection.ObjectType[1]="s_class21";
SharedPools[1].Class[1].PrivateSection.Object[2]="s_v_obj112";
SharedPools[1].Class[1].PrivateSection.ObjectType[2]="class221";
SharedPools[1].Class[1].PrivateSection.Object[3]="s_v_objt113";
SharedPools[1].Class[1].PrivateSection.ObjectType[3]="T9";
SharedPools[1].Class[1].PrivateSection.Object[4]="s_v_obj114";
SharedPools[1].Class[1].PrivateSection.ObjectType[4]="T8";
//SharedCLASS(1,2)
//SharedClass(1,2) public section contains 2 methods and 1 object === VIOLATION
//SMP[1][2]=2 && SOP[1][2]=1  SharedPools[1].Class[2].PublicSection.Method[1].Method-
Name="s_p_mthd1121";
//get ARGS
SharedPools[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgName[1]="A";
SharedPools[1].Class[2].PublicSection.Method[1].ARGS.IN_ArgType[1]="T3";
SharedPools[1].Class[2].PublicSection.Method[1].ReturnedValueType="T4";
//————————————————————
SharedPools[1].Class[2].PublicSection.Method[2].MethodName="s_p_mthd122";
//get ARGS
SharedPools[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgName[1]="A";
SharedPools[1].Class[2].PublicSection.Method[2].ARGS.IN_ArgType[1]="class212";
SharedPools[1].Class[2].PublicSection.Method[2].ReturnedValueType="T4";
//————————————————————
```

```
  SharedPools[1].Class[2].PublicSection.Object[1]="s_p_obj121";
  SharedPools[1].Class[2].PublicSection.ObjectType[1]="class212";//SharedClass(1,2) protected section contains 1 method and 2 objects
//SMR[1][2]=1 && SOR[1][2]=2
  SharedPools[1].Class[2].ProtectedSection.Method[1].MethodName="s_r_mthd121";
  //get ARGS
  SharedPools[1].Class[2].ProtectedSection.Method[1].ReturnedValueType="T4";
//————————————
  SharedPools[1].Class[2].ProtectedSection.Object[1]="s_r_obj121";
  SharedPools[1].Class[2].ProtectedSection.ObjectType[1]="s_class12";
  SharedPools[1].Class[2].ProtectedSection.Object[2]="s_r_obj122";
  SharedPools[1].Class[2].ProtectedSection.ObjectType[2]="T5";//SharedClass(1,2) private section contains 1 method and 4 objects
//SMV[1][2]=1 && SOV[1][2]=4
  SharedPools[1].Class[2].PrivateSection.Method[1].MethodName="s_v_mthd121";
  //get ARGS
  SharedPools[1].Class[2].PrivateSection.Method[1].ReturnedValueType="T3";
//————————————
  SharedPools[1].Class[2].PrivateSection.Object[1]="s_v_obj121";
  SharedPools[1].Class[2].PrivateSection.ObjectType[1]="s_class11";
  SharedPools[1].Class[2].PrivateSection.Object[2]="s_v_obj122";
  SharedPools[1].Class[2].PrivateSection.ObjectType[2]="T6";
  SharedPools[1].Class[2].PrivateSection.Object[3]="s_v_obj123";
  SharedPools[1].Class[2].PrivateSection.ObjectType[3]="T7";
  SharedPools[1].Class[2].PrivateSection.Object[4]="s_v_obj124";
  SharedPools[1].Class[2].PrivateSection.ObjectType[4]="T1";

//SharedCLASS(2,1)
//SharedClass(2,1) public section contains 2 methods and 1 object — VIOLATION
//SMP[2][1]=2 && SOP[2][1]=1
  SharedPools[2].Class[1].PublicSection.Method[1].MethodName="s_p_mthd211";
// get ARGS
  SharedPools[2].Class[1].PublicSection.Method[1].ReturnedValueType="T8";
//————————————
  SharedPools[2].Class[1].PublicSection.Method[2].MethodName="s_p_mthd212";
// get ARGS
  SharedPools[2].Class[1].PublicSection.Method[2].ReturnedValueType="T5";
//————————————
  SharedPools[2].Class[1].PublicSection.Object[1]="s_p_obj211";
  SharedPools[2].Class[1].PublicSection.ObjectType[1]="T12";//SharedClass(2,1) protected section contains 1 method and 2 objects
//SMR[2][1]=1 && SOR[2][1]=2
  SharedPools[2].Class[1].ProtectedSection.Method[1].MethodName="s_r_mthd211";
// get ARGS
  SharedPools[2].Class[1].ProtectedSection.Method[1].ReturnedValueType="T1";
//————————————
  SharedPools[2].Class[1].ProtectedSection.Object[1]="s_r_obj211";
  SharedPools[2].Class[1].ProtectedSection.ObjectType[1]="s_class12";
  SharedPools[2].Class[1].ProtectedSection.Object[2]="s_r_obj212";
  SharedPools[2].Class[1].ProtectedSection.ObjectType[2]="T13";
//SharedClass(2,1) private section contains 1 method and 4 objects
//MV[2][1][1]=1 && OV[2][1][1]=4
  SharedPools[2].Class[1].PrivateSection.Method[1].MethodName="s_v_mthd211";
// get ARGS
  SharedPools[2].Class[1].PrivateSection.Method[1].ReturnedValueType="T2";
//————————————
  SharedPools[2].Class[1].PrivateSection.Object[1]="s_v_obj211";
  SharedPools[2].Class[1].PrivateSection.ObjectType[1]="s_class21";
```

```
SharedPools[2].Class[1].PrivateSection.Object[2]="s_v_obj212";
SharedPools[2].Class[1].PrivateSection.ObjectType[2]="class22";
SharedPools[2].Class[1].PrivateSection.Object[3]="s_v_obj213";
SharedPools[2].Class[1].PrivateSection.ObjectType[3]="T7";
SharedPools[2].Class[1].PrivateSection.Object[4]="s_v_obj214";
SharedPools[2].Class[1].PrivateSection.ObjectType[4]="T6";
//SharedCLASS(2,2)
//SharedClass(2,2) public section contains 2 methods and no object
//SMP[2][2]=2 && SOP[2][2]=0
SharedPools[2].Class[2].PublicSection.Method[1].MethodName="s_p_mthd221";
// get ARGS
//————————————————————
SharedPools[2].Class[2].PublicSection.Method[2].MethodName="s_p_mthd222";
// get ARGS
SharedPools[2].Class[2].PublicSection.Method[2].ReturnedValueType="T3";
//————————————————————//SharedClass(2,2) protected section contains 1 method and 2 objects
//SMR[2][2]=1 && SOR[2][2]=2
SharedPools[2].Class[2].ProtectedSection.Method[1].MethodName="s_r_mthd221";
// get ARGS
SharedPools[2].Class[2].ProtectedSection.Method[1].ReturnedValueType="T2";
//————————————————————
SharedPools[2].Class[2].ProtectedSection.Object[1]="s_r_obj221";
SharedPools[2].Class[2].ProtectedSection.ObjectType[1]="T1";
SharedPools[2].Class[2].ProtectedSection.Object[2]="s_r_obj222";
SharedPools[2].Class[2].ProtectedSection.ObjectType[2]="T2";
//SharedClass(2,2) private section contains 1 method and 4 objects
//SMV[2][2]=1 && SOV[2][2]=4
SharedPools[2].Class[2].PrivateSection.Method[1].MethodName="s_v_mthd221";
// get ARGS
SharedPools[2].Class[2].PrivateSection.Method[1].ReturnedValueType="T7";
//————————————————————
SharedPools[2].Class[2].PrivateSection.Object[1]="s_v_obj221";
SharedPools[2].Class[2].PrivateSection.ObjectType[1]="s_class12";
SharedPools[2].Class[2].PrivateSection.Object[2]="s_v_obj222";
SharedPools[2].Class[2].PrivateSection.ObjectType[2]="s_class21";
SharedPools[2].Class[2].PrivateSection.Object[3]="s_v_obj223";
SharedPools[2].Class[2].PrivateSection.ObjectType[3]="T6";
SharedPools[2].Class[2].PrivateSection.Object[4]="s_v_obj224";
SharedPools[2].Class[2].PrivateSection.ObjectType[4]="T9";//
*****************************************************************
// STEP#4: read in the inheritance references (CLassIDs) for each cless
// *****************************************************************
//SharedClass(1,1) inherits from two other classes in public —
SharedPools[1].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
SharedPools[1].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=2;
SharedPools[1].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;
SharedPools[1].Class[1].PublicInheritedClasses[1]=new ClassID();
SharedPools[1].Class[1].PublicInheritedClasses[1].ClassNodeNumber=1;
SharedPools[1].Class[1].PublicInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
SharedPools[1].Class[1].PublicInheritedClasses[1].ParentComponent.ParentFrameworkNodeNumber=1;

// Class(1,1) inherits from another class in private — Violation — non-public inheritance
SharedPools[1].Class[1].PrivateInheritedSharedClasses[1]=new SharedClassID();
SharedPools[1].Class[1].PrivateInheritedSharedClasses[1].ClassNodeNumber=1;
SharedPools[1].Class[1].PrivateInheritedSharedClasses[1].ParentSharedPoolNodeNumber=1;//     Class(1,2) does not inherit from any other class//Class(2,1) inherits only from one other class in public
```

```
  SharedPools[2].Class[1].PublicInheritedSharedClasses[1]=new SharedClassID();
  SharedPools[2].Class[1].PublicInheritedSharedClasses[1].ClassNodeNumber=1;
  SharedPools[2].Class[1].PublicInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;//Class(2,2) in-
hirets only from one another class in protected
  SharedPools[2].Class[2].ProtectedInheritedSharedClasses[1]=new SharedClassID();
  SharedPools[2].Class[2].ProtectedInheritedSharedClasses[1].ClassNodeNumber=2;
  SharedPools[2].Class[2].ProtectedInheritedSharedClasses[1].ParentSharedPoolNodeNumber=2;//Class(2,2)
inhirets another class in protected from frameworks
  SharedPools[2].Class[2].ProtectedInheritedClasses[1]=new ClassID();
  SharedPools[2].Class[2].ProtectedInheritedClasses[1].ClassNodeNumber=1;
  SharedPools[2].Class[2].ProtectedInheritedClasses[1].ParentComponent.ComponentNodeNumber=2;
  SharedPools[2].Class[2].ProtectedInheritedClasses[1].ParentComponent.ParentFrameworkNodeNum-
ber=2;}
protected static void initializeSharedPools_Example_3_ACE() {
// ****************************************
// STEP#1 read values for:
//      F, C[], c[][],
//      MP[][][], MR[][][], MV[][][]
//      OP[][][], OR[][][], OV[][][]
// ****************************************** //number of shared pools
  S=1;
//number of shared classes per shared pool
  SC[1]=1; //number of public methods per class per shared pool
  SMP[1][1]=0;
//number of protected methods per class per shared pool
  SMR[1][1]=0;
//number of private methods per class per shared pool
  SMV[1][1]=0;
//number of public objects per class per shared pool
  SOP[1][1]=0;
//number of protected methods per class  per shared pool
  SOR[1][1]=0;
//number of private methods per class per shared pool
  SOV[1][1]=0;
//number of classes inherited in public per class per shared pool
  SMinhP[1][1]=0;
//number of classes inherited in protected per class per shared pool
  SMinhR[1][1]=0;
//number of classes inherited in private per class per shared pool
  SMinhV[1][1]=0;
//number of classes inherited in public from frameworks per class per shared pool
  SMinhPF[1][1]=0;
//number of classes inherited in protected from frameworks per class per shared pool
  SMinhRF[1][1]=0;
//number of classes inherited in private from frameworks per class per shared pool
  SMinhVF[1][1]=0;//*********************************************************
// STEP#2: instantiate for the Frameworks
// objects and the embedied
// Class, PublicSection, ProtectedSection, PrivateSection,
// Method, etc. etc, objects
// ALSO: Assign ID values for each object
// *******************************************************// instantiate the framework objects
  for (int i=1; i<=S; i++){
    SharedPools[i]=new SharedPoolNode();
    SharedPools[i].SharedPoolNodeNumber=i;
    //instantiate the shared classes objects for this new shared pool object
    for (int j=1; j<=SC[i]; j++){
      SharedPools[i].Class[j]=new SharedClassNode();
```

```
      SharedPools[i].Class[j].ClassName="dummy_shared_class"+i+j;
      SharedPools[i].Class[j].ID.ClassNodeNumber=j;
      SharedPools[i].Class[j].ID.ParentSharedPoolNodeNumber=i;
      //instantiate for the public section of this class
      SharedPools[i].Class[j].PublicSection=new SharedPublicSectionNode();
      for (int k=1; k<=SMP[i][j];k++){
        SharedPools[i].Class[j].PublicSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PublicSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PublicSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].PublicSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
      }
      //instantiate for the protected section of this class
      SharedPools[i].Class[j].ProtectedSection=new SharedProtectedSectionNode();
      for (int k=1; k<=SMR[i][j];k++){
        SharedPools[i].Class[j].ProtectedSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].ProtectedSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=j;
        SharedPools[i].Class[j].ProtectedSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNode-
Number=i;
      }
      //instantiate for the private section of this class
      SharedPools[i].Class[j].PrivateSection=new SharedPrivateSectionNode();
      for (int k=1; k<=SMV[i][j];k++){
        SharedPools[i].Class[j].PrivateSection.Method[k]=new SharedMethodNode();
        SharedPools[i].Class[j].PrivateSection.Method[k].ReturnedValueType="void";
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.MethodNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ClassNodeNumber=k;
        SharedPools[i].Class[j].PrivateSection.Method[k].ID.ParentSharedClass.ParentSharedPoolNodeNum-
ber=i;
      }
    }
  }
}
/**
 * This method was created by a SmartGuide.
 */
private static void reportStructure (String output) {
System.out.print(output);
return;
}
}
```

# Appendix B2

# A Prototype For CO–Toolkits' Design and Implementation Structures: Example#1 – An Artificial Example

## B2.1 Overview

This Appendix presents the first artificial example that has been used to demonstrate our prototype for CO–Toolkits' design and implementation structures. Section B2.3 shows that this example passes the prototype test (i.e. each predicate returns STATUS=true).

## B2.2 Example#1: Design and Implementation Structure

```
Design and Implementation Structure: As Extracted From The Current Example
Framework(1)
    Component(1,1)
        Class(1,1,1): class111
            Public Inheritance From Classes:        Class(1,2,2):class122
            Public Section Methods & Objects
                MethodName: p_mthd1111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     X(T1)   Y(T2)
                    OUT_ArgNames(Types):    Z(T3)
                MethodName: p_mthd1112
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):     A(T3)   B(T2)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd1111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     X(T2)
                    OUT_ArgNames(Types):    Y(T3)
                Objects: r_obj1112(class112)  r_obj1112(T12)
            Private Section Methods & Objects
                MethodName: v_mthd1111
```

ReturnedValueType: void
IN_ArgNames(Types): Z(T1)
OUT_ArgNames(Types): W(T5)
Objects: v_obj1111(class221) v_obj1112(class222) v_objt1113(T9) v_obj1114(T5)
Class(1,1,2): class112
Public Section Methods & Objects
MethodName: p_mthd1121
ReturnedValueType: T4
IN_ArgNames(Types): A(T3)
OUT_ArgNames(Types):
MethodName: p_mthd1122
ReturnedValueType: T4
IN_ArgNames(Types): A(T3) B(T2)
OUT_ArgNames(Types):
Protected Section Methods & Objects
MethodName: r_mthd1121
ReturnedValueType: T4
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: r_obj1121(class212) r_obj1122(T5)
Private Section Methods & Objects
MethodName: v_mthd1121
ReturnedValueType: T3
IN_ArgNames(Types): A(T3)
OUT_ArgNames(Types):
Objects: v_obj1121(class111) v_obj1122(T6) v_obj1123(T7) v_obj1124(T1)
Component(1,2)
Class(1,2,1): class121
Public Inheritance From Classes: Class(1,2,2):class122
Public Section Methods & Objects
MethodName: p_mthd1211
ReturnedValueType: void
IN_ArgNames(Types):
OUT_ArgNames(Types):
MethodName: p_mthd1212
ReturnedValueType: void
IN_ArgNames(Types): X(T1)
OUT_ArgNames(Types): Z(T3)
Protected Section Methods & Objects
MethodName: r_mthd1211
ReturnedValueType: T3
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: r_obj1211(class212) r_obj1212(T10)
Private Section Methods & Objects
MethodName: v_mthd1211
ReturnedValueType: T2
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: v_obj1211(class212) v_obj1212(T6) v_obj1213(T2) v_obj1214(class222)
Class(1,2,2): class122
Public Section Methods & Objects
MethodName: p_mthd1221
ReturnedValueType: T2
IN_ArgNames(Types): Y(T2)
OUT_ArgNames(Types):
MethodName: p_mthd1222
ReturnedValueType: T1
IN_ArgNames(Types): Y(T1)
OUT_ArgNames(Types):
Protected Section Methods & Objects
MethodName: r_mthd1221
ReturnedValueType: T1
IN_ArgNames(Types): X(T4)
OUT_ArgNames(Types):

```
                Objects: r_obj1221(T11)  r_obj1222(T6)
            Private Section Methods & Objects
                MethodName: v_mthd1221
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects: v_obj1221(class111)  v_obj1222(class221)  v_obj1223(T12)  v_obj1224(T8)
Framework(2)
    Component(2,1)
        Class(2,1,1): class211
            Public Inheritance From Classes:      Class(2,1,2):class212
            Public Section Methods & Objects
                MethodName: p_mthd2111
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):     Z(T6)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2112
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):     W(T1)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd2111
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects: r_obj2111(class112)  r_obj2112(T13)
            Private Section Methods & Objects
                MethodName: v_mthd2111
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects: v_obj2111(class121)  v_obj2112(class122)  v_obj2113(T7)   v_obj2114(T6)
        Class(2,1,2): class212
            Public Section Methods & Objects
                MethodName: p_mthd2121
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     Z(T1)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2122
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):     W(T2)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd2121
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects: r_obj2121(T1)   r_obj2122(T2)
            Private Section Methods & Objects
                MethodName: v_mthd2121
                    ReturnedValueType:      T7
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects: v_obj2121(class212)  v_obj2122(class221)  v_obj2123(class222)  v_obj2124(class211)
    Component(2,2)
        Class(2,2,1): class221
            Public Inheritance From Classes:      Class(2,2,2):class222
            Public Section Methods & Objects
                MethodName: p_mthd2211
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     Z(T1)   Z(T1)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2212
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
```

OUT_ArgNames(Types):
Protected Section Methods & Objects
MethodName: r_mthd2211
ReturnedValueType: T5
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: r_obj2211(T9) r_obj2212(T3)
Private Section Methods & Objects
MethodName: v_mthd2211
ReturnedValueType: T5
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: v_obj2211(T8) v_obj2212(T10) v_obj2213(class211) v_obj2214(class221)
Class(2,2,2): class222
Public Section Methods & Objects
MethodName: p_mthd2221
ReturnedValueType: T2
IN_ArgNames(Types): X(T3)
OUT_ArgNames(Types):
MethodName: p_mthd2222
ReturnedValueType: T8
IN_ArgNames(Types): T(T5)
OUT_ArgNames(Types):
Protected Section Methods & Objects
MethodName: r_mthd2221
ReturnedValueType: T4
IN_ArgNames(Types): R(T5)
OUT_ArgNames(Types):
Objects: r_obj2221(T11) r_obj2222(T21)
Private Section Methods & Objects
MethodName: v_mthd2211
ReturnedValueType: T5
IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: v_obj2221(T7) v_obj2222(class122) v_obj2223(class121) v_obj2224(class222)
SharedPool(1)
Class(1,1): s_class11
Public Inheritance From Classes: SharedClass(1,2)
Public Section Methods & Objects
MethodName: s_p_mthd111
ReturnedValueType: T5
IN_ArgNames(Types): X(class112) Y(s_class12)
OUT_ArgNames(Types):
MethodName: s_p_mthd112
ReturnedValueType: T7
IN_ArgNames(Types): A(T1) B(T2)
OUT_ArgNames(Types):
Objects:
Protected Section Methods & Objects
MethodName: s_r_mthd111
ReturnedValueType: void
IN_ArgNames(Types): X(T7)
OUT_ArgNames(Types): Y(T2)
Objects: s_r_obj112(class112) s_r_obj112(T2)
Private Section Methods & Objects
MethodName: s_v_mthd111
ReturnedValueType: void
IN_ArgNames(Types): W(T6)
OUT_ArgNames(Types): G(s_class22)
Objects: s_v_obj111(s_class21) s_v_obj112(class221) s_v_objt113(T9) s_v_obj114(T8)
Class(1,2): s_class12
Public Section Methods & Objects
MethodName: s_p_mthd1121
ReturnedValueType: T4
IN_ArgNames(Types): A(T3)

```
                OUT_ArgNames(Types):
            MethodName: s_p_mthd122
                ReturnedValueType:      T4
                IN_ArgNames(Types):     A(T5)
                OUT_ArgNames(Types):
            Objects: s_p_obj121(T5)
        Protected Section Methods & Objects
            MethodName: s_r_mthd121
                ReturnedValueType:      T4
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_r_obj121(s_class12)    s_r_obj122(T5)
        Private Section Methods & Objects
            MethodName: s_v_mthd121
                ReturnedValueType:      T3
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_v_obj121(s_class11)    s_v_obj122(T6)  s_v_obj123(T7)  s_v_obj124(T1)
SharedPool(2)
    Class(2,1): s_class21
        Public Inheritance From Classes:      SharedClass(2,2)
        Public Section Methods & Objects
            MethodName: s_p_mthd211
                ReturnedValueType:      T8
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            MethodName: s_p_mthd212
                ReturnedValueType:      T5
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_p_obj211(T12)
        Protected Section Methods & Objects
            MethodName: s_r_mthd211
                ReturnedValueType:      T1
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_r_obj211(s_class12)    s_r_obj212(T13)
        Private Section Methods & Objects
            MethodName: s_v_mthd211
                ReturnedValueType:      T2
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_v_obj211(s_class21)    s_v_obj212(class22)  s_v_obj213(T7)  s_v_obj214(T6)
    Class(2,2): s_class22
        Public Section Methods & Objects
            MethodName: s_p_mthd221
                ReturnedValueType:      void
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            MethodName: s_p_mthd222
                ReturnedValueType:      T3
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:
        Protected Section Methods & Objects
            MethodName: s_r_mthd221
                ReturnedValueType:      T2
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects: s_r_obj221(T1)  s_r_obj222(T2)
        Private Section Methods & Objects
            MethodName: s_v_mthd221
                ReturnedValueType:      T7
```

IN_ArgNames(Types):
OUT_ArgNames(Types):
Objects: s_v_obj221(s_class12) s_v_obj222(s_class21) s_v_obj223(T6) s_v_obj224(T9)

# B2.3 Example#1: Output Report

Checking for Main predicate5_18: GoodDesignAndImplementation
Checking for Predicate5_19: GoodUseOfRelationsAmongComponents
Checking for predicate5_20: SafeInheritance
Checking for predicate 5_23: InheritanceMayNotExistAcrossFrameworks
Checking for predicate5_23 is complete. STATUS: true
Checking for predicate5_24: SafeInheritance_AmongComponents_WithInTheSameFramework
Checking for predicate5_31: A_ComponentMayInheritFromOnlyOneOtherComponentFromTheSameFramework
Checking for predicate5_31 is complete. STATUS: true
Checking for predicate5_32: OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework
Checking for predicate5_32 is complete. STATUS: true
Checking for predicate5_24 is complete. STATUS: true
Checking for Predicate5_25: SafeInheritance_AmongClasses_WithInTheSameComponent
Checking for Predicate 5_33: NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent
Checking for predicate5_33 is complete. STATUS: true
Checking for Predicate5_34: InheritanceMayBe_Public/Protected/or/Private_WithInTheSameComponent
Checking for Predicate5_34 is complete. STATUS: true
Checking for Predicate5_25 is complete. STATUS: true
Checking for predicate 5_26: SafeInheritance_AmongClasses_WithInTheSharedPools
Checking for predicate 5_35: InheritanceMayNotExistAcrossSharedPools
Checking for Predicate5_35 is complete. STATUS: true
Checking for predicate 5_36: NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool
Checking for predicate 5_36 is complete. STATUS: true
Checking for predicate 5_26 is complete. STATUS: true
Checking for predicate5_20 is complete. STATUS: true
Checking for predicate5_21: GoodAssociation
Checking for Predicate 5_27: GoodAssociationOnTheComponentLevel
Checking for predicate5_37: ComponentsMayHaveOnlyUniDirectionalAssociationWithClassesFromTheSharedPools
Checking for predicate5_37 is complete. STATUS: true
Checking for predicate5_38: AssociationMayNotExistAcrossFrameworks
Checking for predicate5_38 is complete. STATUS: true
Checking for Predicate 5_27 is complete. STATUS: true
Checking for Predicate 5_28: GoodAssociationOnInternalClassesLevel
Checking for predicate5_38:AssociationAndInheritanceMayNotExistBetweenTwoClassesAtTheSameTime
Checking for predicate5_38 is complete. STATUS: true
Checking for Predicate 5_28 is complete. STATUS: true
Checking for predicate5_21 is complete. STATUS: true
Checking for predicate5_22: GoodUseOfOtherRelations
Checking for predicate5_22 is complete. STATUS: true
Checking for Predicate5_19 is complete. STATUS: true
Checking for Main predicate5_18: GoodDesignAndImplementation is complete. STATUS: true

# Appendix B3

# A Prototype For CO–Toolkits' Design and Implementation Structures: Example#2 – An Artificial Example

## B3.1 Overview

This Appendix presents the second artificial example that has been used to demonstrate our prototype for CO–Toolkits' design and implementation structures. Section B3.3 presents the rules violations reported by this prototype for this example.

## B3.2 Example#2: Design and Implementation Structure

```
Design and Implementation Structure: As Extracted From The Current Example
Framework(1)
    Component(1,1)
        Class(1,1,1): class111
            Public Inheritance From Classes:        Class(1,2,2):class122 Class(2,2,1):class221
            Protected Inheritance From Classes:     Class(2,1,2):class212
            Private Inheritance From Classes:       Class(2,1,1):class211 SharedClass(2,1):s_class21
            Public Section Methods & Objects
                MethodName: p_mthd1111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     X(T1)     Y(T2)
                    OUT_ArgNames(Types):    Z(T3)
                MethodName: p_mthd1112
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):     A(T3)     B(T2)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd1111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     X(T2)
                    OUT_ArgNames(Types):    Y(T3)
                Objects:   r_obj1112(class112) r_obj1112(T12)
            Private Section Methods & Objects
                MethodName: v_mthd1111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     Z(T1)
                    OUT_ArgNames(Types):    W(T5)
```

```
                Objects:  v_obj1111(class221)  v_obj1112(class222)  v_objt1113(T9)  v_obj1114(T5)
        Class(1,1,2): class112
            Public Section Methods & Objects
                MethodName: p_mthd1121
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):     A(T3)
                    OUT_ArgNames(Types):
                MethodName: p_mthd1122
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):     A(T3)    B(T2)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd1121
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  r_obj1121(class212)  r_obj1122(T5)
            Private Section Methods & Objects
                MethodName: v_mthd1121
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):     A(T3)
                    OUT_ArgNames(Types):
                Objects:  v_obj1121(class111)  v_obj1122(T6)  v_obj1123(T7)  v_obj1124(T1)
    Component(1,2)
        Class(1,2,1): class121
            Public Inheritance From Classes:        Class(1,2,1):class121
            Public Section Methods & Objects
                MethodName: p_mthd1211
                    ReturnedValueType:      void
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                MethodName: p_mthd1212
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     X(T1)
                    OUT_ArgNames(Types):    Z(T3)
            Protected Section Methods & Objects
                MethodName: r_mthd1211
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  r_obj1211(class212)  r_obj1212(T10)
            Private Section Methods & Objects
                MethodName: v_mthd1211
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj1211(class212)  v_obj1212(T6)  v_obj1213(T2)  v_obj1214(class222)
        Class(1,2,2): class122
            Protected Inheritance From Classes:     Class(2,2,1):class221
            Public Section Methods & Objects
                MethodName: p_mthd1221
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):     Y(T2)
                    OUT_ArgNames(Types):
                MethodName: p_mthd1222
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     Y(T1)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd1221
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     X(T4)
                    OUT_ArgNames(Types):
                Objects:  r_obj1221(T11) r_obj1222(T6)
            Private Section Methods & Objects
                MethodName: v_mthd1221
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj1221(class111)  v_obj1222(class221)  v_obj1223(T12)     v_obj1224(T8)
Framework(2)
    Component(2,1)
        Class(2,1,1): class211
            Public Inheritance From Classes:        Class(2,1,1):class211
            Public Section Methods & Objects
                MethodName: p_mthd2111
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):     Z(T6)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2112
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):     W(T1)
                    OUT_ArgNames(Types):
```

```
            Protected Section Methods & Objects
                MethodName: r_mthd2111
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  r_obj2111(class112)  r_obj2112(T13)
            Private Section Methods & Objects
                MethodName: v_mthd2111
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj2111(class121)  v_obj2112(class122)  v_obj2113(T7)  v_obj2114(T6)
        Class(2,1,2): class212
            Public Section Methods & Objects
                MethodName: p_mthd2121
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     Z(T1)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2122
                    ReturnedValueType:      T3
                    IN_ArgNames(Types):     W(T2)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd2121
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  r_obj2121(T1)  r_obj2122(T2)
            Private Section Methods & Objects
                MethodName: v_mthd2121
                    ReturnedValueType:      T7
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj2121(class212)  v_obj2122(class221)  v_obj2123(class222)  v_obj2124(class211)
    Component(2,2)
        Class(2,2,1): class221
            Public Inheritance From Classes:        Class(2,2,2):class222 SharedClass(2,1):s_class21
            Public Section Methods & Objects
                MethodName: p_mthd2211
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):     Z(T1)     Z(T1)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2212
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd2211
                    ReturnedValueType:      T5
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  r_obj2211(T9)  r_obj2212(T3)
            Private Section Methods & Objects
                MethodName: v_mthd2211
                    ReturnedValueType:      T5
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj2211(T8)  v_obj2212(T10)      v_obj2213(class211)  v_obj2214(class221)
        Class(2,2,2): class222      SharedClass(1,2):s_class12
            Public Section Methods & Objects
                MethodName: p_mthd2221
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):     X(T3)
                    OUT_ArgNames(Types):
                MethodName: p_mthd2222
                    ReturnedValueType:      T8
                    IN_ArgNames(Types):     T(T5)
                    OUT_ArgNames(Types):
            Protected Section Methods & Objects
                MethodName: r_mthd2221
                    ReturnedValueType:      T4
                    IN_ArgNames(Types):     R(T5)
                    OUT_ArgNames(Types):
                Objects:  r_obj2221(T11) r_obj2222(T21)
            Private Section Methods & Objects
                MethodName: v_mthd2211
                    ReturnedValueType:      T5
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:  v_obj2221(T7)  v_obj2222(class122)  v_obj2223(class121)  v_obj2224(class222)
Framework(3)
    Component(3,1)
        Class(3,1,1): class311
```

```
            Public Inheritance From Classes:        Class(2.2.1):class221 SharedClass(1.1):s_class11
            Protected Inheritance From Classes:     Class(1.1.2):class112 SharedClass(2.2):s_class22
            Private Inheritance From Classes:       Class(2.2.1):class221 SharedClass(1.2):s_class12
            Public Section Methods & Objects
                MethodName: p_mthd3111
                    ReturnedValueType:      void
                    IN_ArgNames(Types):     W(T4)
                    OUT_ArgNames(Types):    Z(T3)
            Protected Section Methods & Objects
                MethodName: r_mthd3111
                    ReturnedValueType:      T1
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:   r_obj3111(class212)
            Private Section Methods & Objects
                MethodName: v_mthd3111
                    ReturnedValueType:      T2
                    IN_ArgNames(Types):
                    OUT_ArgNames(Types):
                Objects:   v_obj3111(class122)
SharedPool(1)
    Class(1.1): s_class11
        Public Inheritance From Classes:        SharedClass(2.2)
    Class(1.2.1)
        Private Inheritance From Classes:       SharedClass(1.1)
        Public Section Methods & Objects
            MethodName: s_p_mthd111
                ReturnedValueType:      T5
                IN_ArgNames(Types):     X(class112)    Y(s_class12)
                OUT_ArgNames(Types):
            MethodName: s_p_mthd112
                ReturnedValueType:      T7
                IN_ArgNames(Types):     A(T1)    B(T2)
                OUT_ArgNames(Types):
            Objects:
        Protected Section Methods & Objects
            MethodName: s_r_mthd111
                ReturnedValueType:      void
                IN_ArgNames(Types):     X(T7)
                OUT_ArgNames(Types):    Y(T2)
            Objects:   s_r_obj112(class112) s_r_obj112(T2)
        Private Section Methods & Objects
            MethodName: s_v_mthd111
                ReturnedValueType:      void
                IN_ArgNames(Types):     W(T6)
                OUT_ArgNames(Types):    G(s_class22)
            Objects:   s_v_obj111(s_class21)    s_v_obj112(class221)    s_v_objt113(T9)    s_v_obj114(T8)
    Class(1.2): s_class12
        Public Section Methods & Objects
            MethodName: s_p_mthd1121
                ReturnedValueType:      T4
                IN_ArgNames(Types):     A(T3)
                OUT_ArgNames(Types):
            MethodName: s_p_mthd122
                ReturnedValueType:      T4
                IN_ArgNames(Types):     A(class212)
                OUT_ArgNames(Types):
            Objects:   s_p_obj121(class212)
        Protected Section Methods & Objects
            MethodName: s_r_mthd121
                ReturnedValueType:      T4
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_r_obj121(s_class12)    s_r_obj122(T5)
        Private Section Methods & Objects
            MethodName: s_v_mthd121
                ReturnedValueType:      T3
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_v_obj121(s_class11)    s_v_obj122(T6) s_v_obj123(T7) s_v_obj124(T1)
SharedPool(2)
    Class(2.1): s_class21
        Public Inheritance From Classes:        SharedClass(2.1)
        Public Section Methods & Objects
            MethodName: s_p_mthd211
                ReturnedValueType:      T8
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            MethodName: s_p_mthd212
                ReturnedValueType:      T5
```

```
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_p_obj211(T12)
        Protected Section Methods & Objects
            MethodName: s_r_mthd211
                ReturnedValueType:      T1
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_r_obj211(s_class12)      s_r_obj212(T13)
        Private Section Methods & Objects
            MethodName: s_v_mthd211
                ReturnedValueType:      T2
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_v_obj211(s_class21)      s_v_obj212(class22)  s_v_obj213(T7) s_v_obj214(T6)
    Class(2,2): s_class22
        Protected Inheritance From Classes:       SharedClass(2,2)      Class(2,2,1)
        Public Section Methods & Objects
            MethodName: s_p_mthd221
                ReturnedValueType:      void
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            MethodName: s_p_mthd222
                ReturnedValueType:      T3
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:
        Protected Section Methods & Objects
            MethodName: s_r_mthd221
                ReturnedValueType:      T2
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_r_obj221(T1) s_r_obj222(T2)
        Private Section Methods & Objects
            MethodName: s_v_mthd221
                ReturnedValueType:      T7
                IN_ArgNames(Types):
                OUT_ArgNames(Types):
            Objects:   s_v_obj221(s_class12)      s_v_obj222(s_class21)      s_v_obj223(T6) s_v_obj224(T9)
==================================================================
```

# B3.3 Example#2: Output Report

```
Checking for Main predicate5_18: GoodDesignAndImplementation
Checking for Predicate5_19: GoodUseOfRelationsAmongComponents
Checking for predicate5_20: SafeInheritance
Checking for predicate 5_23: InheritanceMayNotExistAcrossFrameworks
VIOLATION(Predicate5_23): Class(1,1,1) inherits in public from Class(2,2,1) from another framework
VIOLATION(Predicate5_23): Class(1,1,1) inherits in protected from Class(2,1,2) from another framework
VIOLATION(Predicate5_23): Class(1,1,1) inherits in private from Class(2,1,1) from another framework
VIOLATION(Predicate5_23): Class(1,2,2) inherits in protected from Class(2,2,1) from another framework
VIOLATION(Predicate5_23): Class(3,1,1) inherits in public from Class(2,2,1) from another framework
VIOLATION(Predicate5_23): Class(3,1,1) inherits in protected from Class(1,1,2) from another framework
VIOLATION(Predicate5_23): Class(3,1,1) inherits in private from Class(2,2,1) from another framework
Checking for predicate5_23 is complete. STATUS: false
Checking for predicate5_24: SafeInheritance_AmongComponents_WithInTheSameFramework
Checking for predicate5_30: A_ComponentMayInheritFromOnlyOneOtherComponentFromTheSameFramework
VIOLATION(Predicate5_30): Class(1,1,1) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_30): Class(1,1,1) inherits from Class(2,1,2) from different framework
VIOLATION(Predicate5_30): Class(1,1,1) inherits from Class(2,1,1) from different framework
VIOLATION(Predicate5_30): Class(1,2,2) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_30): Class(3,1,1) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_30): Class(3,1,1) inherits from Class(1,1,2) from different framework
VIOLATION(Predicate5_30): Class(3,1,1) inherits from Class(2,2,1) from different framework
Checking for predicate5_30 is complete. STATUS: false
Checking for predicate5_31: OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework
VIOLATION(Predicate5_31): Class(1,1,1) inherits from Class(2,1,2) from different framework
VIOLATION(Predicate5_31): Class(1,1,1) inherits from Class(2,1,1) from different framework
VIOLATION(Predicate5_31): Class(1,2,2) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_31): Class(3,1,1) inherits from Class(1,1,2) from different framework
VIOLATION(Predicate5_31): Class(3,1,1) inherits from Class(2,2,1) from different framework
Checking for predicate5_31 is complete. STATUS: false
Checking for predicate5_24 is complete. STATUS: false
```

```
Checking for Predicate5_25: SafeInheritance_AmongClasses_WithInTheSameComponent
Checking for Predicate 5_32: NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent
VIOLATION(Predicate5_32): Class(1,1,1) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_32): Class(1,1,1) inherits from Class(2,1,2) from different framework
VIOLATION(Predicate5_32): Class(1,1,1) inherits from Class(2,1,1) from different framework
VIOLATION(Predicate5_32): Class(1,2,2) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_32): Class(3,1,1) inherits from Class(2,2,1) from different framework
VIOLATION(Predicate5_32): Class(3,1,1) inherits from Class(1,1,2) from different framework
VIOLATION(Predicate5_32): Class(3,1,1) inherits from Class(2,2,1) from different framework
Checking for predicate5_32 is complete. STATUS: false
Checking for Predicate5_33: InheritanceMayBe_Public/Protected/or/Private_WithInTheSameComponent
Checking for Predicate5_33 is complete. STATUS: true
Checking for Predicate5_25 is complete. STATUS: false
Checking for predicate 5_26: SafeInheritance_AmongClasses_WithInTheSharedPools
Checking for predicate 5_34: InheritanceMayNotExistAcrossSharedPools
VIOLATION(Predicate5_34): SharedClass(1,1) inherits from SharedClass(2,2) from another shared pool
Checking for Predicate5_34 is complete. STATUS: false
Checking for predicate 5_35: NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool
VIOLATION(predicate5_35): SharedClass(1,1) inherits from SharedClass(2,2) from another shared pool
Checking for predicate 5_35 is complete. STATUS: false
Checking for predicate 5_26 is complete. STATUS: false
Checking for predicate5_20 is complete. STATUS: false
Checking for predicate5_21: GoodAssociation
Checking for Predicate 5_27: GoodAssociationOnTheComponentLevel
Checking for predicate5_36: ComponentsMayHaveOnlyUniDirectionalAssociationWithClassesFromTheSharedPools
Checking for predicate5_36 is complete. STATUS: true
Checking for predicate5_37: AssociationMayNotExistAcrossFrameworks
Checking for predicate5_37 is complete. STATUS: true
Checking for Predicate 5_27 is complete. STATUS: true
Checking for Predicate 5_28: GoodAssociationOnInternalClassesLevel
Checking for predicate5_38:AssociationAndInheritanceMayNotExistBetweenTwoClassesAtTheSameTime
Checking for predicate5_38 is complete. STATUS: true
Checking for Predicate 5_28 is complete. STATUS: true
Checking for predicate5_21 is complete. STATUS: true
Checking for predicate5_22: GoodUseOfOtherRelations
Checking for predicate5_22 is complete. STATUS: true
Checking for Predicate5_19 is complete. STATUS: false
Checking for Main predicate5_18: GoodDesignAndImplementation   is complete. STATUS: false
```

# Appendix B4

# A Prototype For CO–Toolkits' Design and Implementation Structures: Example#3 – ACE Sample Inheritance Structure

## B4.1 Overview

This Appendix presents the third example that has been used to demonstrate our prototype for CO–Toolkits' design and implementation structures. This example was based on a sample ACE inheritance structure. Section B4.3 presents the rules violations reported by this prototype for this example.

## B4.2 Example#3: ACE Sample Design and Implementation Structure

```
Design and Implementation Structure: As Extracted From The Current Example
Framework(1)
    Component(1,1)
        Class(1,1,1): ACE_Module_Link
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(1,2)
        Class(1,2,1): ACE_Map_Manager
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(1,2,2): ACE_Map_Iterator
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
```

Private Section Methods & Objects
Objects:
Class(1,2,3): ACE_Map_Reverse_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,3)
Class(1,3,1): ACE_Message_Block
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,4)
Class(1,4,1): ACE_Notification_Strategy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,4,2): ACE_Message_Queue
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,4,3): ACE_Message_Queue_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,4,4): ACE_Message_Queue_Reverse_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,5)
Class(1,5,1): ACE_Module
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,6)
Class(1,6,1): Driver
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,6,2): ACE_Multiplexor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,7)
Class(1,7,1): ACE_Stream
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,7,2): ACE_Stream_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(1,8)
Class(1,8,1): ACE_Stream_Head
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(1,8,2): ACE_Stream_Tail
Public Section Methods & Objects

                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(1,8,3): ACE_Thru_Task
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(1,9)
            Class(1,9,1): ACE_Task_Flags
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(1,9,2): ACE_Task_Base
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(1,9,3): ACE_Task_Exit
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
    Framework(2)
        Component(2,1)
            Class(2,1,1): ACE_CORBA_Handler
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(2,1,2): ACE_ST_CORBA_Handler
                Public Inheritance From Classes:        Class(2,1,1):ACE_CORBA_Handler
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(2,1,3): ACE_MT_CORBA_Handler
                Public Inheritance From Classes:        Class(2,1,1):ACE_CORBA_Handler
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(2,2)
            Class(2,2,1): ACE_CORBA_Ref
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
    Framework(3)
        Component(3,1)
            Class(3,1,1): ACE_Set_Node
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,1,2): ACE_Unbounded_Set_Iterator
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,1,3): ACE_Set_Node
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,1,4): ACE_Unbounded_Set
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:

Private Section Methods & Objects
Objects:
Class(3.1.5): ACE_Fixed_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.1.6): ACE_Fixed_Set_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.1.7): ACE_Fixed_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.1.8): ACE_Bounded_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.1.9): ACE_Bounded_Set_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.1.10): ACE_Bounded_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(3.2)
Class(3.2.1): ACE_Stack_Node
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.2): ACE_Queue_Node
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.3): ACE_Bounded_Stack
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.4): ACE_Fixed_Stack
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.5): ACE_Stack_Node
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.6): ACE_Unbounded_Stack
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.7): ACE_Queue_Node
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(3.2.8): ACE_Unbounded_Queue

```
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(3,3)
            Class(3,3,1): ACE_Allocator
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,3,2): ACE_CString
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,3,3): ACE_SString
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(3,3,4): ACE_WString
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
    Framework(4)
        Component(4,1)
            Class(4,1,1): ACE_Activation_Queue
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(4,2)
            Class(4,2,1): ACE_Method_Object
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(4,3)
            Class(4,3,1): ACE_Process_Descriptor
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(4,3,2): ACE_Process_Control
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(4,3,3): ACE_Process_Manager
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(4,3,4): ACE_Process_Control
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
        Component(4,4)
            Class(4,4,1): ACE_File_Lock
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
                    Objects:
            Class(4,4,2): ACE_Semaphore
                Public Section Methods & Objects
                Protected Section Methods & Objects
                    Objects:
                Private Section Methods & Objects
```

Objects:
Class(4,4,3): ACE_Process_Semaphore
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,4): ACE_RW_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,5): ACE_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,6): ACE_Process_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,7): ACE_RW_Process_Mutex
Public Inheritance From Classes: Class(4,4,6):ACE_Process_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,8): ACE_Null_Barrier
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,9): ACE_Null_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,10): ACE_Null_Condition_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,11): ACE_Null_Mutex_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,12): ACE_Event
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,13): ACE_Manual_Event
Public Inheritance From Classes: Class(4,4,12):ACE_Event
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,14): ACE_Auto_Event
Public Inheritance From Classes: Class(4,4,12):ACE_Event
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,4,15): ACE_Thread_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:

| | |
|---|---|
| Class(4,4,16): ACE_Thread_Mutex_Guard | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,17): ACE_Condition_Thread_Mutex | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,18): ACE_Recursive_Thread_Mutex | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,19): ACE_RW_Thread_Mutex | |
| Public Inheritance From Classes: | Class(4,4,4):ACE_RW_Mutex |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,20): ACE_Thread_Semaphore | |
| Public Inheritance From Classes: | Class(4,4,2):ACE_Semaphore |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,21): ACE_Barrier | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,22): ACE_Process_Barrier | |
| Public Inheritance From Classes: | Class(4,4,21):ACE_Barrier |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,4,23): ACE_Thread_Barrier | |
| Public Inheritance From Classes: | Class(4,4,21):ACE_Barrier |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Component(4,5) | |
| Class(4,5,1): ACE_Synch_Options | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Component(4,6) | |
| Class(4,6,1): ACE_Test_and_Set | |
| Public Inheritance From Classes: | Class(12,1,1):ACE_Event_Handler |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,6,2): ACE_Atomic_Op | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,6,3): ACE_TSS | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |
| Objects: | |
| Private Section Methods & Objects | |
| Objects: | |
| Class(4,6,4): ACE_NULL_SYNCH | |
| Public Section Methods & Objects | |
| Protected Section Methods & Objects | |

Objects:
Private Section Methods & Objects
Objects:
Class(4.6.5): ACE_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.6): ACE_Write_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.7): ACE_Read_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.8): ACE_TSS_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.9): ACE_TSS_Write_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.10): ACE_TSS_Read_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.11): ACE_Condition
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.12): ACE_Process_Condition
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.13): ACE_Thread_Condition
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.6.14): ACE_MT_SYNCH
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(4.7)
Class(4.7.1): ACE_Thread
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(4.8)
Class(4.8.1): ACE_Task_Base
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4.8.2): ACE_Thread_Manager
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects

Objects:
Class(4,8,3): ACE_Thread_Descriptor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,8,4): ACE_Thread_Control
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,8,5): ACE_Thread_Manager
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(4,8,6): ACE_Thread_Control
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(4,9)
Class(4,9,1): ACE_Token
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(5)
Component(5,1)
Class(5,1,1): ACE_Acceptor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,1,2): ACE_Strategy_Acceptor
Public Inheritance From Classes: Class(5,1,1):ACE_Acceptor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,1,3): ACE_Oneshot_Acceptor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(5,2)
Class(5,2,1): ACE_Svc_Tuple
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,2,2): ACE_Connector
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(5,3)
Class(5,3,1): ACE_Dynamic_Service
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(5,4)
Class(5,4,1): ACE_Reactor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,4,2): ACE_ReactorEx
Public Section Methods & Objects

Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,4,3): ACE_Notification_Strategy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,4,4): ACE_Reactor_Notification_Strategy
Public Inheritance From Classes: Class(5,4,3):ACE_Notification_Strategy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(5,4,5): ACE_ReactorEx_Notification_Strategy
Public Inheritance From Classes: Class(5,4,3):ACE_Notification_Strategy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(5,5)
Class(5,5,1): ACE_Svc_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(6)
Component(6,1)
Class(6,1,1): ACE_IO_SAP
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,2)
Class(6,2,1): ACE_DEV
Public Inheritance From Classes: Class(6,1,1):ACE_IO_SAP
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,3)
Class(6,3,1): ACE_DEV_Connector
Public Inheritance From Classes: Class(6,2,1):ACE_DEV
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,4)
Class(6,4,1): ACE_DEV_IO
Public Inheritance From Classes: Class(6,2,1):ACE_DEV
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,5)
Class(6,5,1): ACE_FILE
Public Inheritance From Classes: Class(6,1,1):ACE_IO_SAP
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,6)
Class(6,6,1): ACE_FILE_Connector
Public Inheritance From Classes: Class(6,5,1):ACE_FILE
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,7)
Class(6,7,1): ACE_FILE_IO
Public Inheritance From Classes: Class(6,5,1):ACE_FILE

- Public Section Methods & Objects
- Protected Section Methods & Objects
  - Objects:
- Private Section Methods & Objects
  - Objects:

Component(6,8)
- Class(6,8,1): ACE_IPC_SAP
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,9)
- Class(6,9,1): ACE_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,10)
- Class(6,10,1): ACE_DEV_Addr
  - Public Inheritance From Classes: Class(6,10,1):ACE_DEV_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,11)
- Class(6,11,1): ACE_FILE_Addr
  - Public Inheritance From Classes: Class(6,11,1):ACE_FILE_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,12)
- Class(6,12,1): ACE_INET_Addr
  - Public Inheritance From Classes: Class(6,9,1):ACE_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,13)
- Class(6,13,1): ACE_SPIPE_Addr
  - Public Inheritance From Classes: Class(6,9,1):ACE_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,14)
- Class(6,14,1): ACE_UNIX_Addr
  - Public Inheritance From Classes: Class(6,9,1):ACE_Addr
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,15)
- Class(6,15,1): ACE_FIFO
  - Public Inheritance From Classes: Class(6,15,1):ACE_FIFO
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,16)
- Class(6,16,1): ACE_FIFO_Recv
  - Public Inheritance From Classes: Class(6,15,1):ACE_FIFO
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects
    - Objects:

Component(6,17)
- Class(6,17,1): ACE_FIFO_Recv_Msg
  - Public Inheritance From Classes: Class(6,16,1):ACE_FIFO_Recv
  - Public Section Methods & Objects
  - Protected Section Methods & Objects
    - Objects:
  - Private Section Methods & Objects

Objects:
Component(6,18)
Class(6,18,1): ACE_FIFO_Send
Public Inheritance From Classes: Class(6,15,1):ACE_FIFO
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,19)
Class(6,19,1): ACE_FIFO_Send_Msg
Public Inheritance From Classes: Class(6,18,1):ACE_FIFO_Send
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,20)
Class(6,20,1): ACE_LSOCK
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,21)
Class(6,21,1): ACE_LSOCK_Acceptor
Public Inheritance From Classes: Class(6,26,1):ACE_SOCK_Acceptor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,22)
Class(6,22,1): ACE_LSOCK_CODgram
Public Inheritance From Classes: Class(6,27,1):ACE_SOCK_CODgram Class(6,20,1):ACE_LSOCK
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,23)
Class(6,23,1): ACE_LSOCK_Connector
Public Inheritance From Classes: Class(6,28,1):ACE_SOCK_Connector
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,24)
Class(6,24,1): ACE_LSOCK_Dgram
Public Inheritance From Classes: Class(6,29,1):ACE_SOCK_Dgram Class(6,20,1):ACE_LSOCK
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(6,24,2): ACE_LSOCK_Stream
Public Inheritance From Classes: Class(6,33,1):ACE_SOCK_Stream Class(6,20,1):ACE_LSOCK
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,25)
Class(6,25,1): ACE_SOCK
Public Inheritance From Classes: Class(6,8,1):ACE_IPC_SAP
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,26)
Class(6,26,1): ACE_SOCK_Acceptor
Public Inheritance From Classes: Class(6,25,1):ACE_SOCK
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(6,27)
Class(6,27,1): ACE_SOCK_CODgram
Public Inheritance From Classes: Class(6,32,1):ACE_SOCK_IO

```
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.28)
        Class(6.28.1): ACE_SOCK_Connector
            Public Inheritance From Classes:       Class(6.25.1):ACE_SOCK
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.29)
        Class(6.29.1): ACE_SOCK_Dgram
            Public Inheritance From Classes:       Class(6.25.1):ACE_SOCK
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.30)
        Class(6.30.1): ACE_SOCK_Dgram_Bcast
            Public Inheritance From Classes:       Class(6.29.1):ACE_SOCK_Dgram
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.31)
        Class(6.31.1): ACE_SOCK_Dgram_Mcast
            Public Inheritance From Classes:       Class(6.29.1):ACE_SOCK_Dgram
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.32)
        Class(6.32.1): ACE_SOCK_IO
            Public Inheritance From Classes:       Class(6.25.1):ACE_SOCK
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.33)
        Class(6.33.1): ACE_SOCK_Stream
            Public Inheritance From Classes:       Class(6.32.1):ACE_SOCK_IO
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.34)
        Class(6.34.1): ACE_SPIPE
            Public Inheritance From Classes:       Class(6.8.1):ACE_IPC_SAP
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.35)
        Class(6.35.1): ACE_SPIPE_Acceptor
            Public Inheritance From Classes:       Class(6.34.1):ACE_SPIPE
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.36)
        Class(6.36.1): ACE_SPIPE_Connector
            Public Inheritance From Classes:       Class(6.34.1):ACE_SPIPE
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6.37)
        Class(6.37.1): ACE_SPIPE_Stream
            Public Inheritance From Classes:       Class(6.34.1):ACE_SPIPE
            Public Section Methods & Objects
            Protected Section Methods & Objects
```

```
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,38)
        Class(6,38,1): ACE_TLI
            Public Inheritance From Classes:        Class(6,8,1):ACE_IPC_SAP
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,39)
        Class(6,39,1): ACE_TLI_Request_Queue
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(6,39,2): ACE_TLI_Request_Queue
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(6,39,3): ACE_TLI_Acceptor
            Public Inheritance From Classes:        Class(6,38,1):ACE_TLI
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,40)
        Class(6,40,1): ACE_TLI_Connector
            Public Inheritance From Classes:        Class(6,38,1):ACE_TLI
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,41)
        Class(6,41,1): ACE_TLI_Stream
            Public Inheritance From Classes:        Class(6,38,1):ACE_TLI
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,42)
        Class(6,42,1): ACE_UPIPE_Acceptor
            Public Inheritance From Classes:        Class(6,35,1):ACE_SPIPE_Acceptor
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,43)
        Class(6,43,1): ACE_UPIPE_Connector
            Public Inheritance From Classes:        Class(6,34,1):ACE_SPIPE
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(6,44)
        Class(6,44,1): ACE_UPIPE_Stream
            Public Inheritance From Classes:        Class(6,34,1):ACE_SPIPE
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
Framework(7)
    Component(7,1)
        Class(7,1,1): ACE_Log_Msg
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(7,2)
        Class(7,2,1): ACE_Log_Record
            Public Section Methods & Objects
```

Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(8)
Component(8,1)
Class(8,1,1): ACE_Mem_Map
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(8,2)
Class(8,2,1): ACE_Allocator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,2,2): ACE_Name_Node
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,2,3): ACE_Control_Block
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(8,3)
Class(8,3,1): ACE_Allocator_Adapter
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,3,2): ACE_Malloc
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,3,3): ACE_Malloc_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(8,4)
Class(8,4,1): ACE_Sbrk_Memory_Pool_Options
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,4,2): ACE_Sbrk_Memory_Pool
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,4,3): ACE_Shared_Memory_Pool_Options
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,4,4): ACE_Shared_Memory_Pool
Public Inheritance From Classes: Class(12,1,1):ACE_Event_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(8,4,5): ACE_Local_Memory_Pool_Options
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:

- Class(8,4,6): ACE_Local_Memory_Pool
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
        - Objects:
    - Private Section Methods & Objects
        - Objects:
- Class(8,4,7): ACE_MMAP_Memory_Pool_Options
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
        - Objects:
    - Private Section Methods & Objects
        - Objects:
- Class(8,4,8): ACE_MMAP_Memory_Pool
    - Public Inheritance From Classes: Class(12,1,1):ACE_Event_Handler
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
        - Objects:
    - Private Section Methods & Objects
        - Objects:
- Class(8,4,9): ACE_Lite_MMAP_Memory_Pool
    - Public Inheritance From Classes: Class(8,4,9):ACE_Lite_MMAP_Memory_Pool
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
        - Objects:
    - Private Section Methods & Objects
        - Objects:

- Component(8,5)
    - Class(8,5,1): ACE_Shared_Memory
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
- Component(8,6)
    - Class(8,6,1): ACE_Shared_Memory_MM
        - Public Inheritance From Classes: Class(8,5,1):ACE_Shared_Memory
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
- Component(8,7)
    - Class(8,7,1): ACE_Shared_Memory_SV
        - Public Inheritance From Classes: Class(8,5,1):ACE_Shared_Memory
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:

Framework(9)

- Component(9,1)
    - Class(9,1,1): auto_ptr
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
    - Class(9,1,2): auto_array_ptr
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
- Component(9,2)
    - Class(9,2,1): ACE_Dynamic
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
- Component(9,3)
    - Class(9,3,1): ACE_Get_Opt
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:
        - Private Section Methods & Objects
            - Objects:
- Component(9,4)
    - Class(9,4,1): ACE_Singleton
        - Public Section Methods & Objects
        - Protected Section Methods & Objects
            - Objects:

Private Section Methods & Objects
Objects:
Framework(10)
Component(10,1)
Class(10,1,1): ACE_NS_String
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(10,1,2): ACE_NS_Internal
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,2)
Class(10,2,1): ACE_Name_Space_Map
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(10,2,2): ACE_Local_Name_Space
Public Inheritance From Classes: Class(10,5,2):ACE_Name_Space
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,3)
Class(10,3,1): ACE_Name_Proxy
Public Inheritance From Classes: Class(12,1,1):ACE_Event_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,4)
Class(10,4,1): ACE_Name_Request
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(10,4,2): ACE_Name_Reply
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,5)
Class(10,5,1): ACE_Name_Binding
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(10,5,2): ACE_Name_Space
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,6)
Class(10,6,1): ACE_Naming_Context
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(10,6,2): ACE_Name_Options
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(10,7)
Class(10,7,1): ACE_Remote_Name_Space
Public Inheritance From Classes: Class(10,5,2):ACE_Name_Space
Public Section Methods & Objects
Protected Section Methods & Objects

- Objects:
- Private Section Methods & Objects
  - Objects:

Framework(11)
- Component(11.1)
  - Class(11.1.1): ACE_TSS_Ref
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(11.1.2): ACE_TSS_Info
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(11.1.3): ACE_TSS_Cleanup
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(11.1.4): ACE_Thread_Adapter
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(11.1.5): ACE_OS
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:

Framework(12)
- Component(12.1)
  - Class(12.1.1): ACE_Event_Handler
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
- Component(12.2)
  - Class(12.2.1): ACE_Event_Handler_T
    - Public Inheritance From Classes: Class(12.1.1):ACE_Event_Handler
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
- Component(12.3)
  - Class(12.3.1): ACE_Handle_Set
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(12.3.2): ACE_Handle_Set_Iterator
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
- Component(12.4)
  - Class(12.4.1): ACE_Overlapped_IO
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(12.4.2): ACE_Proactor
    - Public Inheritance From Classes: Class(12.1.1):ACE_Event_Handler
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:
    - Private Section Methods & Objects
      - Objects:
  - Class(12.4.3): ACE_Overlapped_File
    - Public Section Methods & Objects
    - Protected Section Methods & Objects
      - Objects:

Private Section Methods & Objects
Objects:
Class(12.4.4): ACE_Reactor_Notify
Public Inheritance From Classes: Class(12.1.1):ACE_Event_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.4.5): ACE_Reactor_Token
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.4.6): ACE_Handler_Repository
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.4.7): ACE_Handler_Repository_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.4.8): ACE_Reactor
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(12.5)
Class(12.5.1): ACE_ReactorEx_Token
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.5.2): ACE_ReactorEx_Notify
Public Inheritance From Classes: Class(12.1.1):ACE_Event_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.5.3): ACE_ReactorEx
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(12.6)
Class(12.6.1): ACE_Sig_Handlers_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.6.2): ACE_Sig_Set
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.6.3): ACE_Sig_Action
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.6.4): ACE_Sig_Guard
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.6.5): ACE_Sig_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:

Private Section Methods & Objects
Objects:
Class(12.6.6): ACE_Sig_Adapter
Public Inheritance From Classes: Class(12.1.1):ACE_Event_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.6.7): ACE_Sig_Handlers
Public Inheritance From Classes: Class(12.6.5):ACE_Sig_Handler
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(12.7)
Class(12.7.1): ACE_Time_Value
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(12.7.2): ACE_Countdown_Time
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(12.8)
Class(12.8.1): ACE_Timer_Queue
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(13)
Component(13.1)
Class(13.1.1): ACE_SV_Message
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.2)
Class(13.2.1): ACE_SV_Message_Queue
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.3)
Class(13.3.1): ACE_Typed_SV_Message
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.4)
Class(13.4.1): ACE_Typed_SV_Message_Queue
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.5)
Class(13.5.1): ACE_SV_Semaphore_Complex
Public Inheritance From Classes: Class(13.6.1):ACE_SV_Semaphore_Simple
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.6)
Class(13.6.1): ACE_SV_Semaphore_Simple
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(13.7)
Class(13.7.1): ACE_SV_Shared_Memory

Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(14)
Component(14.1)
Class(14.1.1): ACE_Profile_Timer
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(14.1.2): ACE_Profile_Timer
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(14.2)
Class(14.2.1): ACE_High_Res_Timer
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Framework(15)
Component(15.1)
Class(15.1.1): ACE_TOKEN_CONST
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.2): ACE_TPQ_Entry
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.3): ACE_TSS_TPQ_Entry
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.4): ACE_TPQ_Iterator
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.5): ACE_Token_Proxy_Queue
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.6): ACE_Tokens
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.7): ACE_Local_Mutex
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.8): ACE_Mutex_Token
Public Inheritance From Classes: Class(15.1.6):ACE_Tokens
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15.1.9): ACE_RW_Token
Public Inheritance From Classes: Class(15.1.6):ACE_Tokens
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:

Private Section Methods & Objects
Objects:
Class(15,1,10): ACE_Token_Name
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,1,11): ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,1,12): ACE_Null_Token
Public Inheritance From Classes: Class(15,1,11):ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,1,13): ACE_Local_Mutex
Public Inheritance From Classes: Class(15,1,11):ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,1,14): ACE_Local_RLock
Public Inheritance From Classes: Class(15,1,11):ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,1,15): ACE_Local_WLock
Public Inheritance From Classes: Class(15,1,11):ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(15,2)
Class(15,2,1): ACE_Token_Name
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Component(15,3)
Class(15,3,1): ACE_Remote_Token_Proxy
Public Inheritance From Classes: Class(15,1,11):ACE_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,3,2): ACE_Remote_Mutex
Public Inheritance From Classes: Class(15,3,1):ACE_Remote_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,3,3): ACE_Remote_RLock
Public Inheritance From Classes: Class(15,3,1):ACE_Remote_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,3,4): ACE_Remote_WLock
Public Inheritance From Classes: Class(15,3,1):ACE_Remote_Token_Proxy
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:
Private Section Methods & Objects
Objects:
Class(15,3,5): ACE_TSS_Connection
Public Section Methods & Objects
Protected Section Methods & Objects
Objects:

```
            Private Section Methods & Objects
                Objects:
    Component(15,4)
        Class(15,4,1): ACE_Token_Collection
            Public Inheritance From Classes:        Class(15,1,11):ACE_Token_Proxy
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(15,5)
        Class(15,5,1): ACE_Token_Manager
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(15,6)
        Class(15,6,1): ACE_Mutex_Invariants
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(15,6,2): ACE_RWLock_Invariants
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(15,6,3): ACE_Token_Invariant_Manager
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
    Component(15,7)
        Class(15,7,1): ACE_Token_Request
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
        Class(15,7,2): ACE_Token_Reply
            Public Section Methods & Objects
            Protected Section Methods & Objects
                Objects:
            Private Section Methods & Objects
                Objects:
SharedPool(1)
    Class(1,1): dummy_shared_class11
        Public Section Methods & Objects
            Objects:
        Protected Section Methods & Objects
            Objects:
        Private Section Methods & Objects
            Objects:
=====================================================================
```

# B4.3 Example#3: ACE Sample Output Report

```
Checking for Main predicate5_18: GoodDesignAndImplementation
Checking for Predicate5_19: GoodUseOfRelationsAmongComponents
Checking for predicate5_20: SafeInheritance
Checking for predicate 5_23: InheritanceMayNotExistAcrossFrameworks
VIOLATION(Predicate5_23): Class(4,6,1) inherits in public from Class(12,1,1) from another framework
VIOLATION(Predicate5_23): Class(8,4,4) inherits in public from Class(12,1,1) from another framework
VIOLATION(Predicate5_23): Class(8,4,8) inherits in public from Class(12,1,1) from another framework
VIOLATION(Predicate5_23): Class(10,3,1) inherits in public from Class(12,1,1) from another framework
Checking for predicate5_23 is complete. STATUS: false
Checking for predicate5_24: SafeInheritance_AmongComponents_WithInTheSameFramework
Checking for predicate5_30: A_ComponentMayInheritFromOnlyOneOtherComponentFromTheSameFramework
VIOLATION(Predicate5_30): Class(4,6,1) inherits from Class(12,1,1) from different framework
```

```
VIOLATION(Predicate5_30): Class(8,4,4) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_30): Class(8,4,8) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_30): Class(10,3,1) inherits from Class(12,1,1) from different framework
Checking for predicate5_30 is complete. STATUS: false
Checking for predicate5_31: OnlyPublicInheritanceMayExistAcrossComponentsWithInTheSameFramework
Checking for predicate5_31 is complete. STATUS: true
Checking for predicate5_24 is complete. STATUS: false
Checking for Predicate5_25: SafeInheritance_AmongClasses_WithInTheSameComponent
Checking for Predicate 5_32: NoMultipleInheritanceMayExistAmongClassesWithInTheSameComponent
VIOLATION(Predicate5_32: Class(2,1,3) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(4,4,13) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(4,4,19) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(4,4,22) has multiple inheritance within the same component
VIOLATION(Predicate5_32): Class(4,6,1) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_32: Class(5,4,5) has multiple inheritance within the same component
VIOLATION(Predicate5_32): Class(8,4,4) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_32): Class(8,4,8) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_32): Class(10,3,1) inherits from Class(12,1,1) from different framework
VIOLATION(Predicate5_32: Class(15,1,9) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(15,1,13) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(15,1,15) has multiple inheritance within the same component
VIOLATION(Predicate5_32: Class(15,3,3) has multiple inheritance within the same component
Checking for predicate5_32 is complete. STATUS: false
Checking for Predicate5_33: InheritanceMayBe_Public/Protected/or/Private_WithInTheSameComponent
Checking for Predicate5_33 is complete. STATUS: true
Checking for Predicate5_25 is complete. STATUS: false
Checking for predicate 5_26: SafeInheritance_AmongClasses_WithInTheSharedPools
Checking for predicate 5_34: InheritanceMayNotExistAcrossSharedPools
Checking for Predicate5_34 is complete. STATUS: true
Checking for predicate 5_35: NoMultipleInheritanceMayExistAmongClassesWithInTheSameSharedPool
Checking for predicate 5_35 is complete. STATUS: true
Checking for predicate 5_26 is complete. STATUS: true
Checking for predicate5_20 is complete. STATUS: false
Checking for predicate5_21: GoodAssociation
Checking for predicate5_21 is complete. STATUS: true
Checking for predicate5_22: GoodUseOfOtherRelations
Checking for predicate5_22 is complete. STATUS: true
Checking for Predicate5_19 is complete. STATUS: false
Checking for Main predicate5_18: GoodDesignAndImplementation   is complete. STATUS: false
```

# APPENDIX C

# SAMPLE DOCUMENTS FOR THE DOCCUMENTATION VIEW

*Appendix C: Prototype: View 3 – Documentation View*

# Appendix C1

# Sample CGD Tables

This appendix presents sample tables that can be contained in a CGD. The following table that can be used as a quick reference for finding what components are in each framework as well as what files should be included in the user's application such that it becomes possible to use a certain component. According to Schmidt in his report as the external examiner for this thesis, this kind of documents can take advantage of hyperlinks.

**Table C1–1: List of ACE frameworks and their components**

| | *FRAMEWORK* | *SOURCE–FILE* | *COMPONENT* | *BASE–CLASS (Component)* |
|---|---|---|---|---|
| 1 | [ASX] | IO_Cntl_Msg.h | ACE_IO_Cntl_Msg | N/D |
| 2 | | IO_Cntl_Msg.cpp | ACE_Module_Link | N/D |
| 3 | | Map_Manager.h | ACE_Allocator | N/D |
| 4 | | | ACE_Map_Iterator | N/D |
| 5 | | | ACE_Map_Manager | N/D |
| 6 | | Message_Block.h | ACE_Message_Block | N/D |
| 7 | | Message_Queue.h | ACE_Message_Queue | N/D |
| 8 | | Module.h | ACE_Module | N/D |
| 9 | | Multiplexor.h | Driver | N/D |
| 10 | | | ACE_Multiplexor | N/D |
| 11 | | Stream.h | ACE_Stream | N/D |
| 12 | | | ACE_Stream_Iterator | N/D |
| 13 | | Stream_Modules.h | ACE_Stream_Head | public ACE_Task<ACE_SYNCH_2> |
| 14 | | | ACE_Stream_Tail | public ACE_Task<ACE_SYNCH_2> |
| 15 | | | ACE_Thru_Task | public ACE_Task<ACE_SYNCH_2> |
| 16 | | Task.h | ACE_Task_Flags | N/D |
| 17 | | | ACE_Task | public ACE_Service_Object |
| 18 | | | ACE_Task_Exit | N/D |

**Table C1-1: List of ACE frameworks and their components (part 2)**

| | *FRAMEWORK* | *SOURCE-FILE* | *COMPONENT* | *BASE-CLASS (Component)* |
|---|---|---|---|---|
| 19 | [CORBA] | CORBA_Handler.h | ACE_CORBA_Handler | public ACE_Service_Objec |
| 20 | | | ACE_ST_CORBA_Handler | public ACE_CORBA_Handler |
| 21 | | | ACE_MT_CORBA_Handler | public ACE_CORBA_Handler, public CORBA_1 (ThreadFilter) |
| 22 | | CORBA_Ref.h | ACE_CORBA_Ref | N/D |
| 23 | [Collections] | Set.h | ACE_Unbounded_Set_Iterator | N/D |
| 24 | | | ACE_Set_Node | N/D |
| 25 | | | ACE_Unbounded_Set | N/D |
| 26 | | | ACE_Fixed_Set | N/D |
| 27 | | | ACE_Fixed_Set_Iterator | N/D |
| 28 | | | ACE_Bounded_Set | N/D |
| 29 | | | ACE_Bounded_Set_Iterator | N/D |
| 30 | | Set.cpp | ACE_Set_Node | N/D |
| 31 | | Stack.h | ACE_Bounded_Stack | N/D |
| 32 | | | ACE_Fixed_Stack | N/D |
| 33 | | | ACE_Stack_Node | N/D |
| 34 | | | ACE_Unbounded_Stack | N/D |
| 35 | | | ACE_Queue_Node | N/D |
| 36 | | | ACE_Unbounded_Queue | N/D |
| 37 | | Stack.cpp | ACE_Stack_Node | N/D |
| 38 | | | ACE_Queue_Node | N/D |
| 39 | | SString.h | ACE_Allocator | N/D |
| 40 | | | ACE_CString | N/D |
| 41 | | | ACE_SString | N/D |
| 42 | | | ACE_WString | N/D |
| 43 | [Concurrency] ⇓ | Process_Manager.h | ACE_Process_Control | N/D |
| 44 | | | ACE_Process_Manager | N/D |
| 45 | | | ACE_Process_Control | N/D |
| 46 | | | ACE_Process_Descriptor | N/D |
| 47 | | Synch.h ⇓ | ACE_File_Lock | N/D |
| 48 | | | ACE_Time_Value | N/D |
| 49 | | | ACE_Mutex | N/D |
| 50 | | | ACE_Thread_Mutex | N/D |
| 51 | | | ACE_Thread_Mutex_Guard | N/D |
| 52 | | | ACE_Condition_Thread_Mutex | N/D |

## Table C1-1: List of ACE frameworks and their components (part 3)

| | FRAMEWORK | SOURCE-FILE | COMPONENT | BASE-CLASS (Component) |
|---|---|---|---|---|
| 53 | [Concurrency] | Synch.h | ACE_Recursive_Thread_Mutex | N/D |
| 54 | | | ACE_RW_Mutex | N/D |
| 55 | | | ACE_RW_Thread_Mutex | public ACE_RW_Mutex |
| 56 | | | ACE_Semaphore | N/D |
| 57 | | | ACE_Process_Semaphore | public ACE_Semaphore |
| 58 | | | ACE_Thread_Semaphore | public ACE_Semaphore |
| 59 | | | ACE_Barrier | N/D |
| 60 | | | ACE_Process_Barrier | public ACE_Barrier |
| 61 | | | ACE_Thread_Barrier | public ACE_Barrier |
| 62 | | | ACE_Process_Mutex | N/D |
| 63 | | | ACE_RW_Process_Mutex | public ACE_Process_Mutex |
| 64 | | | ACE_Null_Mutex | N/D |
| 65 | | | ACE_Null_Condition_Mutex | N/D |
| 66 | | | ACE_Null_Mutex_Guard | N/D |
| 67 | | Synch_Options.h | ACE_Synch_Options | N/D |
| 68 | | Synch_T.h | ACE_Time_Value | N/D |
| 69 | | | ACE_Atomic_Op | N/D |
| 70 | | | ACE_TSS | N/D |
| 71 | | | ACE_NULL_SYNCH | N/D |
| 72 | | | ACE_Guard | N/D |
| 73 | | | ACE_Write_Guard | public ACE_Guard<LOCK> |
| 74 | | | ACE_Read_Guard | public ACE_Guard<LOCK> |
| 75 | | | ACE_TSS_Guard | N/D |
| 76 | | | ACE_TSS_Write_Guard | ACE_TSS_Guard<LOCK> |
| 77 | | | ACE_TSS_Read_Guard | ACE_TSS_Guard<LOCK> |
| 78 | | | ACE_Condition | N/D |
| 79 | | | ACE_Process_Condition | public ACE_Condition<MUTEX> |
| 80 | | | ACE_Thread_Condition | public ACE_Condition<MUTEX> |
| 81 | | | ACE_MT_SYNCH | N/D |
| 82 | | Thread.h | ACE_Thread | N/D |
| 83 | | Thread_Manager.h | ACE_Thread_Manager | N/D |
| 84 | | Thread_Manager.h | ACE_Thread_Descriptor | N/D |
| 85 | | | ACE_Thread_Control | N/D |
| 86 | | Token.h | ACE_Token | N/D |

**Table C1–1: List of ACE frameworks and their components (part 4)**

| | FRAMEWORK | SOURCE–FILE | COMPONENT | BASE–CLASS (Component) |
|---|---|---|---|---|
| 87 | [Connection] | Acceptor.h | ACE_Acceptor | public ACE_Service_Object |
| 88 | | | ACE_Strategy_Acceptor | public ACE_Acceptor <SVC_HANDLER, ACE_PEER_ACCEPTOR_2> |
| 89 | | | ACE_Oneshot_Acceptor | public ACE_Service_Object |
| 90 | | Connector.h | ACE_Svc_Tuple | N/D |
| 91 | | | ACE_Connecto | public ACE_Service_Object |
| 92 | | Dynamic_Service.h | ACE_Dynamic_Service | N/D |
| 93 | | Strategies.h | ACE_Concurrency_Strategy | N/D |
| 94 | | | ACE_Thread_Strategy | public ACE_Concurrency_Strategy<SVC_HANDLER> |
| 95 | | | ACE_Process_Strategy | public ACE_Concurrency_Strategy<SVC_HANDLER> |
| 96 | | | ACE_Accept_Strategy | N/D |
| 97 | | | ACE_Scheduling_Strategy | N/D |
| 98 | | | ACE_Schedule_All_Reactive_Strategy | public ACE_Scheduling_Strategy<SVC_HANDLER> |
| 99 | | | ACE_Schedule_All_Threaded_Strategy | public ACE_Scheduling_Strategy<SVC_HANDLER> |
| 100 | | | ACE_Creation_Strategy | N/D |
| 101 | | | ACE_Singleton_Strategy | public ACE_Creation_Strategy<SVC_HANDLER> |
| 102 | | | ACE_DLL_Strategy | public ACE_Creation_Strategy<SVC_HANDLER> |
| 103 | | Svc_Handler.h | ACE_Dynamic | N/D |
| 104 | | | ACE_Svc_Handler | public ACE_Task<ACE_SYNCH_2> |
| 105 | [Debugging] | Dump.h | ACE_Dumpable | N/D |
| 106 | | | ACE_Dumpable_Ptr | N/D |
| 107 | | | ACE_OD | N/D |
| 108 | | Trace.h | ACE_Trace | N/D |
| 109 | | Dump_T.h | ACE_Dumpable_Adapter | public ACE_Dumpable |
| 110 | | Pipe.h | ACE_Pipe | N/D |
| 111 | [IPC] [IO_SAP] | IO_SAP.h | ACE_IO_SAP | N/D |
| 112 | [IPC] [IO_SAP] [DEV_SAP] | DEV.h | ACE_DEV | public ACE_IO_SAP |
| 113 | | DEV_Connector.h | ACE_DEV_Connector | public ACE_DEV |
| 114 | | DEV_IO.h | ACE_DEV_IO | public ACE_DEV |

**Table C1–1: List of ACE frameworks and their components (part 5)**

| | *FRAMEWORK* | *SOURCE-FILE* | *COMPONENT* | *BASE-CLASS (Component)* |
|---|---|---|---|---|
| 115 | [IPC] [IO_SAP] [FILE_SAP] | FILE.h | ACE_FILE | public ACE_IO_SAP |
| 116 | | FILE_Connector.h | ACE_FILE_Connector | public ACE_FILE |
| 117 | | FILE_IO.h | ACE_FILE_IO | public ACE_FILE |
| 118 | [IPC] [IPC_SAP] | IPC_SAP.h | ACE_IPC_SAP | N/D |
| 119 | [IPC] [IPC_SAP] [Addr] | Addr.h | ACE_Export ACE_Addr | N/D |
| 120 | | DEV_Addr.h | ACE_DEV_Addr | public ACE_Addr |
| 121 | | FILE_Addr.h | ACE_FILE_Addr | public ACE_Addr |
| 122 | | INET_Addr.h | ACE_INET_Addr | public ACE_Addr |
| 123 | | SPIPE_Addr.h | ACE_SPIPE_Addr | public ACE_Addr |
| 124 | | UNIX_Addr.h | ACE_UNIX_Addr | public ACE_Addr |
| 125 | [IPC] [IPC_SAP] [FIFO_SAP] | FIFO.h | ACE_FIFO | public ACE_IPC_SAP |
| 126 | | FIFO_Recv.h | ACE_FIFO_Recv | public ACE_FIFO |
| 127 | | FIFO_Recv_Msg.h | ACE_FIFO_Recv_Msg | public ACE_FIFO_ |
| 128 | | FIFO_Send.h | ACE_FIFO_Send | public ACE_FIFO |
| 129 | | FIFO_Send_Msg.h | ACE_FIFO_Send_Msg | public ACE_FIFO_Send |
| 130 | [IPC] [IPC_SAP] [SOCK_SAP] | LSOCK.h | ACE_LSOCK | N/D |
| 131 | | LSOCK_Acceptor.h | ACE_LSOCK_Acceptor | public ACE_SOCK_Acceptor |
| 132 | | LSOCK_CODgram.h | ACE_LSOCK_CODgram | public ACE_SOCK_CODgram, public ACE_LSOCK |
| 133 | | LSOCK_Connector.h | ACE_LSOCK_Connector | public ACE_SOCK_Connector |
| 134 | | LSOCK_Dgram.h | ACE_LSOCK_Dgram | public ACE_SOCK_Dgram, public ACE_LSOCK |
| 135 | | LSOCK_Stream.h | ACE_LSOCK_Stream | public ACE_SOCK_Stream, public ACE_LSOCK |
| 136 | | SOCK.h | ACE_SOCK | public ACE_IPC_SAP |
| 137 | | SOCK_Acceptor.h | ACE_SOCK_Acceptor | public ACE_SOCK |
| 138 | | SOCK_CODgram.h | ACE_SOCK_CODgram | public ACE_SOCK_IO |
| 139 | | SOCK_Connector.h | ACE_SOCK_Connector | public ACE_SOCK |
| 140 | | SOCK_Dgram.h | ACE_SOCK_Dgram | public ACE_SOCK |
| 141 | | SOCK_Dgram_ Bcast.h | ACE_SOCK_Dgram_Bcast | public ACE_SOCK_Dgram |
| 142 | | SOCK_Dgram_ Mcast.h | ACE_SOCK_Dgram_Mcast | public ACE_SOCK_Dgram |

**Table C1-1: List of ACE frameworks and their components (part 6)**

| | FRAMEWORK | SOURCE-FILE | COMPONENT | BASE-CLASS (Component) |
|---|---|---|---|---|
| 143 | [IPC] [IPC_SAP] [SOCK_SAP] | SOCK_IO.h | ACE_SOCK_IO | public ACE_SOCK |
| 144 | | SOCK_Stream.h | ACE_SOCK_Stream | public ACE_SOCK_IO |
| 145 | [IPC] [IPC_SAP] [SPIPE_SAP] | SPIPE.h | ACE_SPIPE | public ACE_IPC_SAP |
| 146 | | SPIPE_Acceptor.h | ACE_SPIPE_Acceptor | public ACE_SPIPE |
| 147 | | SPIPE_Connector.h | ACE_SPIPE_Connector | public ACE_SPIPE |
| 148 | | SPIPE_Stream.h | ACE_SPIPE_Stream | public ACE_SPIPE |
| 149 | [IPC] [IPC_SAP] [TLI_SAP] | TLI.h | ACE_TLI | public ACE_IPC_SAP |
| 150 | | TLI_Acceptor.h | ACE_TLI_Request_Queue | N/D |
| 151 | | | ACE_TLI_Acceptor | public ACE_TLI |
| 152 | | TLI_Acceptor.cpp | ACE_TLI_Request_Queue | N/D |
| 153 | | TLI_Connector.h | ACE_TLI_Connector | public ACE_TLI |
| 154 | | TLI_Stream.h | ACE_TLI_Stream | public ACE_TLI |
| 155 | [IPC] [UPIPE_SAP] | UPIPE_Acceptor.h | ACE_UPIPE_Acceptor | public ACE_SPIPE_Acceptor |
| 156 | | UPIPE_Connector.h | ACE_UPIPE_Connector | public ACE_SPIPE |
| 157 | | UPIPE_Stream.h | ACE_UPIPE_Stream | public ACE_SPIPE |
| 158 | [Log_Msg] | Log_Msg.h | ACE_Log_Msg | N/D |
| 159 | | Log_Record.h | ACE_Log_Record | N/D |
| 160 | [Memory] [Mem_Map] | Mem_Map.h | ACE_Mem_Map | N/D |
| 161 | [Memory] [Shared_Malloc] | Malloc.h | ACE_Allocator | N/D |
| 162 | | | ACE_Name_Node | N/D |
| 163 | | | ACE_Control_Block | N/D |
| 164 | | Malloc_T.h | ACE_Allocator_Adapter | public ACE_Allocator |
| 165 | | | ACE_Malloc_Iterator | N/D |
| 166 | | | ACE_Malloc | N/D |
| 167 | | Memory_Pool.h | ACE_Sbrk_Memory_Pool | N/D |
| 168 | | | ACE_Shared_Memory_Pool | public ACE_Event_Handler |
| 169 | | | ACE_Local_Memory_Pool | N/D |
| 170 | | | ACE_MMAP_Memory_Pool | public ACE_Event_Handler |
| 171 | [Memory] [Shared_Memory] | Shared_Memory.h | ACE_Shared_Memory | N/D |
| 172 | | Shared_Memory_MM.h | ACE_Shared_Memory_MM | public ACE_Shared_Memory |
| 173 | | Shared_Memory_SV.h | ACE_Shared_Memory_SV | public ACE_Shared_Memory |

**Table C1–1: List of ACE frameworks and their components (part 7)**

| | FRAMEWORK | SOURCE–FILE | COMPONENT | BASE–CLASS (Component) |
|---|---|---|---|---|
| 174 | [Misc] | ARGV.h | ACE_ARGV | N/D |
| 175 | | Auto_Ptr.h | auto_ptr | N/D |
| 176 | | | auto_array_ptr | N/D |
| 177 | | Date_Time.h | ACE_Date_Time | N/D |
| 178 | | Dynamic.h | ACE_Dynamic | N/D |
| 179 | | Get_Opt.h | ACE_Get_Opt | N/D |
| 180 | | Obstack.h | ACE_Obchunk | N/D |
| 181 | | | ACE_Obstack | N/D |
| 182 | | Read_Buffer.h | ACE_Read_Buffer | N/D |
| 183 | | Singleton.h | ACE_Singleton | N/D |
| 184 | [Name_Service] | Local_ Name_Space.h | ACE_Name_Space_Map | N/D |
| 185 | | | ACE_Export ACE_NS_String | N/D |
| 186 | | | ACE_Export ACE_NS_Inter- nal | N/D |
| 187 | | | ACE_Export ACE_ Local_Name_Space | public ACE_Name_Space |
| 188 | | Local_ Name_Space.cpp | ACE_Name_Space_Map | public ACE_Map_Manager <ACE_NS_String, ACE_NS_Internal, ACE_RW_Process_Mutex> |
| 189 | | Name_Proxy.h | ACE_Name_Proxy | public ACE_Event_Handler |
| 190 | | Name_Request_ Reply.h | ACE_Name_Request | N/D |
| 191 | | | ACE_Name_Reply | N/D |
| 192 | | Name_Space.h | ACE_Name_Binding | N/D |
| 193 | | | ACE_Name_Space | N/D |
| 194 | | Naming_Context.h | ACE_Name_Options | N/D |
| 195 | | | ACE_Naming_Context | public ACE_Service_Object |
| 196 | | Remote_ Name_Space.h | ACE_Remote_Name_Space | public ACE_Name_Space |
| 197 | [OS] | ACE.h | ACE_Time_Value | N/D |
| 198 | | | ACE | N/D |
| 199 | | OS.h | ACE_Static_Svc_##X | N/D |
| 200 | | | ACE_OS | N/D |
| 201 | | OS.cpp | ACE_TSS_Ref | N/D |
| 202 | | | ACE_TSS_Info | N/D |
| 203 | | | ACE_TSS_Cleanup | N/D |
| 204 | | | ACE_Win32_Thread_Adapt- er | N/D |

## Table C1–1: List of ACE frameworks and their components (part 8)

| | FRAMEWORK | SOURCE–FILE | COMPONENT | BASE–CLASS (Component) |
|---|---|---|---|---|
| 205 | [Reactor] | Event_Handler.h | ACE_Message_Block | N/D |
| 206 | | | ACE_Event_Handler | N/D |
| 207 | | Event_Handler_T.h | ACE_Event_Handler_T | public ACE_Event_Handler |
| 208 | | Handle_Set.h | ACE_Handle_Set | N/D |
| 209 | | | ACE_Handle_Set_Iterator | N/D |
| 210 | | Proactor.h | ACE_Overlapped_File | N/D |
| 211 | | | ACE_Overlapped_IO | N/D |
| 212 | | | ACE_Proactor | public ACE_Event_Handler |
| 213 | | Proactor.cpp | ACE_Overlapped_IO | public ACE_OVERLAPPED |
| 214 | | Reactor.h | ACE_Reactor | N/D |
| 215 | | | ACE_Notification_Handler | public ACE_Event_Handler |
| 216 | | | ACE_Reactor_Token | public ACE_REACTOR_MUTEX |
| 217 | | | ACE_Handler_Repository | N/D |
| 218 | | | ACE_Handler_Repository_ Iterator | N/D |
| 219 | | Signal.h | ACE_Sig_Set | N/D |
| 220 | | | ACE_Sig_Action | N/D |
| 221 | | | ACE_Sig_Guard | N/D |
| 222 | | | ACE_Sig_Handler | N/D |
| 223 | | | ACE_Sig_Handlers | public ACE_Sig_Handler |
| 224 | | Signal.cpp | ACE_SH | public ACE_Event_Handler |
| 225 | | | ACE_Sig_Handlers_Set | N/D |
| 226 | | Time_Value.h | ACE_Time_Value | N/D |
| 227 | | Timer_Queue.h | ACE_Timer_Queue | N/D |
| 228 | | ReactorEx.h | ACE_ReactorEx | N/D |
| 229 | | | ACE_ReactorEx_Token | public ACE_REACTOREX_MUTEX |
| 230 | [Service_ Configurator] ⇓ | Parse_Node.h | ACE_Parse_Node | N/D |
| 231 | | | ACE_Suspend_Node | public ACE_Parse_Node |
| 232 | | | ACE_Resume_Node | public ACE_Parse_Node |
| 233 | | | ACE_Remove_Node | public ACE_Parse_Node |
| 234 | | | ACE_Static_Node | public ACE_Parse_Node |
| 235 | | Parse_Node.h | ACE_Dynamic_Node | public ACE_Static_Node |
| 236 | | | ACE_Stream_Node | public ACE_Parse_Node |
| 237 | | | ACE_Location_Node | N/D |
| 238 | | | ACE_Object_Node | public ACE_Location_Node |

## Table C1–1: List of ACE frameworks and their components (part 9)

| | *FRAMEWORK* | *SOURCE-FILE* | *COMPONENT* | *BASE-CLASS (Component)* |
|---|---|---|---|---|
| 239 | [Service_ Configurator] | Parse_Node.h | ACE_Function_Node | public ACE_Location_Node |
| 240 | | | ACE_Dummy_Node | public ACE_Parse_Node |
| 241 | | Service_Config.h | ACE_Service_Repository | N/D |
| 242 | | | ACE_Service_Record | N/D |
| 243 | | | ACE_Allocator | N/D |
| 244 | | | ACE_Reactor | N/D |
| 245 | | | ACE_Service_Config | public ACE_Event_Handle |
| 246 | | Service_Manager.h | ACE_Service_Manager | public ACE_Service_Object |
| 247 | | Service_Object.h | ACE_Service_Object | public ACE_Event_Handler, public ACE_Shared_Object |
| 248 | | | ACE_Service_Type | N/D |
| 249 | | Service_Record.h | ACE_Service_Object_Type | public ACE_Service_Type |
| 250 | | | ACE_Module_Type | public ACE_Service_Type |
| 251 | | | ACE_Stream_Type | public ACE_Service_Type |
| 252 | | | ACE_Service_Record | N/D |
| 253 | | Service_Repository.h | ACE_Service_Repository | N/D |
| 254 | | | ACE_Service_Repository_ Iterator | N/D |
| 255 | | Shared_Object.h | ACE_Shared_Object | N/D |
| 256 | [System_V_IPC] [System_V_Mes-sage_Queues] | SV_Message.h | ACE_SV_Message | N/D |
| 257 | | SV_ Message_Queue.h | ACE_SV_Message_Queue | N/D |
| 258 | | Typed_SV_ Message.h | ACE_Typed_SV_Message | N/D |
| 259 | | Typed_SV_ Message_Queue.h | ACE_Typed_SV_ Message_Queue | N/D |
| 260 | [System_V_IPC] [System_V_Sema-phores] | SV_Semaphore_ Complex.h | ACE_SV_ Semaphore_Complex | private ACE_SV_Semaphore_Simple |
| 261 | | SV_Semaphore_ Simple.h | ACE_SV_Semaphore_Simple | N/D |
| 262 | [System_V_IPC] [System_V_ Shared_Memory] | SV_Shared_Memory.h | ACE_SV_Shared_Memory | N/D |
| 263 | [Timers] | High_Res_Timer.h | ACE_High_Res_Timer | N/D |
| 264 | | Profile_Timer.h | ACE_Profile_Timer | N/D |
| 265 | | Time_Request_ Reply.h | ACE_Time_Request | N/D |

## Table C1–1: List of ACE frameworks and their components (part 10)

| | FRAMEWORK | SOURCE-FILE | COMPONENT | BASE-CLASS (Component) |
|---|---|---|---|---|
| 266 | [Timers] | System_Time.h ? | ACE_Date_Time | N/D |
| 267 | | | ACE_System_Time | N/D |
| 268 | [Token_Service] | Local_Tokens.h | ACE_TOKEN_CONST | N/D |
| 269 | | | ACE_Token_Proxy | N/D |
| 270 | | | ACE_TPQ_Entry | N/D |
| 271 | | | ACE_TSS_TPQ_Entry | public ACE_TSS<ACE_TPQ_Entry> |
| 272 | | | ACE_Token_Proxy_Queue | N/D |
| 273 | | | ACE_Export ACE_Tokens | N/D |
| 274 | | | ACE_Local_Mutex | N/D |
| 275 | | | ACE_Mutex_Token | public ACE_Tokens |
| 276 | | | ACE_RW_Token | public ACE_Tokens |
| 277 | | | ACE_Token_Proxy | N/D |
| 278 | | | ACE_Null_Token | public ACE_Token_Proxy |
| 279 | | | ACE_Local_Mutex | public ACE_Token_Proxy |
| 280 | | | ACE_Local_RLock | public ACE_Token_Proxy |
| 281 | | | ACE_Local_WLock | public ACE_Token_Proxy |
| 282 | | Local_Tokens_T.h | ACE_Token_Name | N/D |
| 283 | | Remote_Tokens.h | ACE_Remote_Token_Proxy | public ACE_Token_Proxy |
| 284 | | | ACE_Remote_Mutex | public ACE_Remote_Token_Proxy |
| 285 | | | ACE_Remote_RLock | public ACE_Remote_Token_Proxy |
| 286 | | | ACE_Remote_WLock | public ACE_Remote_Token_Proxy |
| 287 | | | ACE_TSS_Connection | public ACE_TSS<ACE_SOCK_Stream> |
| 288 | | Token_Collection.h | ACE_Token_Collection | public ACE_Token_Proxy |
| 289 | | Token_Manager.h | ACE_Local_Mutex | N/D |
| 290 | | | ACE_Mutex_Token | N/D |
| 291 | | | ACE_Token_Manager | N/D |
| 292 | | Token_Request_Reply.h | ACE_Token_Request | N/D |
| 293 | | | ACE_Token_Reply | N/D |

? : This File is not listed under any category in the original document of the ACE categories. This original document will be included as is in the appendices.

N/D : Not Derived from another class

⇓ : To continue the cell on the next page

The following is a sample table that can be used as a quick reference for identifying what service(s) can be provided by a certain component(s).

**Table C-2: Component-Indexed Table**

| | *COMPONENT NAMES* | *Services* |
|---|---|---|
| 1 | ComponentName1 | Service of Component1 |
| 2 | ComponentName2 | Service of Component2 |
| 3 | ComponentName3 | Service of Component3 |
| | ...... | ....... |
| | ...... | ....... |
| | ...... | ....... |
| | ...... | ....... |
| | ComponentNameN | Service of ComponentN |

The following is a sample table that can be used as a quick reference for identifying which component(s) can provide a certain service(s).

**Table C-3: Service-Indexed Table**

| | *SERVICE NAMES* | *Components* |
|---|---|---|
| 1 | ServiceName1 | Component of Service1 |
| 2 | ServiceName2 | Component of Service2 |
| 3 | ServiceName3 | Component of Service3 |
| | ...... | ....... |
| | ...... | ....... |
| | ...... | ....... |
| | ...... | ....... |
| | ServiceNameN | Component of ServiceN |

# Appendix C2

# Sample CISD Interface Specifications

## C2.1. Overview

A CO–Toolkit's CISD is intended to provide a precise description (specification) for the interface of each component that is provided in that CO–toolkit. This appendix presents a sample interface specifications for one of the many components (i.e. ACE_Message_Block of the ASX framework) found in ACE.

## C2.2. Interface Specification

### C2.2.1. CHARACTERISTICS

***Type Specified:*** ACE_Message_Block

***Public Primitive System Data Types:***

| *Types* | *Informal Description* |
|---|---|
| size_t | sizes of objects (such as strings) (unsigned) |
| u_long | unsigned long |

***InheritedPublicPrimitive System Data Types:*** None

***Public Specification Constant Parameters:*** The following table shows all the specification parameters that are used in this class;

<table>
<tr><th>Parameter</th><th colspan="2">Informal Description</th></tr>
<tr><td>#DONT_DELETE#</td><td>Don't delete the data on exit since we don't own it</td><td rowspan="2">Constant parameters</td></tr>
<tr><td>#USER_FLAGS#</td><td>user defined flags start here</td></tr>
<tr><td>#MB_DATA#</td><td>regular data</td><td rowspan="2">Data and protocol messages (regular and priority)</td></tr>
<tr><td>#MB_PROTO#</td><td>protocol control</td></tr>
<tr><td>#MB_BREAK#</td><td>line break</td><td rowspan="6">Control messages (regular and priority)</td></tr>
<tr><td>#MB_PASSFP#</td><td>pass file pointer</td></tr>
<tr><td>#MB_EVENT#</td><td>post an event to an event queue</td></tr>
<tr><td>#MB_SIG#</td><td>generate process signal</td></tr>
<tr><td>#MB_IOCTL#</td><td>ioctl; set/get params</td></tr>
<tr><td>#MB_SETOPTS#</td><td>set various stream head options</td></tr>
<tr><td>#MB_IOCACK#</td><td>acknowledge ioctl</td><td rowspan="11">Control messages (high priority; go to head of queue)</td></tr>
<tr><td>#MB_IOCNAK#</td><td>negative ioctl acknowledge</td></tr>
<tr><td>#MB_PCPROTO#</td><td>priority proto message</td></tr>
<tr><td>#MB_PCSIG#</td><td>generate process signal</td></tr>
<tr><td>#MB_READ#</td><td>generate read notification</td></tr>
<tr><td>#MB_FLUSH#</td><td>flush your queues</td></tr>
<tr><td>#MB_STOP#</td><td>stop transmission immediately</td></tr>
<tr><td>#MB_START#</td><td>restart transmission after stop</td></tr>
<tr><td>#MB_HANGUP#</td><td>line disconnect</td></tr>
<tr><td>#MB_ERROR#</td><td>fatal error used to set u.u_error</td></tr>
<tr><td>#MB_PCEVENT#</td><td>post an event to an event queue</td></tr>
<tr><td>#MB_NORMAL#</td><td>Normal priority messages</td><td rowspan="2">Message class masks</td></tr>
<tr><td>#MB_PRIORITY#</td><td>High priority control messages</td></tr>
</table>

## *Public Specification Types:*

ACE_Message_Type: enum = { #MB_DATA# , #MB_PROTO#, #MB_BREAK#, #MB_PASSFP#, #MB_EVENT#, #MB_SIG#, #MB_IOCTL#, #MB_SETOPTS#, #MB_IOCACK#, #MB_IOCNAK#, #MB_PCPROTO#, #MB_PCSIG#, #MB_READ#, #MB_FLUSH#, #MB_STOP#, #MB_START#, #MB_HANGUP#, #MB_ERROR#, #MB_PCEVENT#, #MB_NORMAL# , #MB_PRIORITY# }

Message_Flags: u_long

## *Public Specification Variables:* None

# C2.2.2. SYNTAX

The Declaration of this class is in the Message_Block.h header file, while the implementations of its access programs are in the Message_Block.i and Message_Block.cpp header files.

## C2.2.2.1. Public ACCESS PROGRAMS

| Program Name | Value | Arg #1 | Arg #2 | Arg #3 | Arg #4 | Arg #5 | *Informal Description* | *Header File* |
|---|---|---|---|---|---|---|---|---|
| ACE_Message_Block | | | | | | | *Creates an empty message* | *Message_Block.cpp* |
| ACE_Message_Block | char* | char* | | | | | *Creates a Message Block that assumes ownership of <data> (i.e., doesn't delete it since it didn't malloc it!)* | *Message_Block.cpp* |
| ACE_Message_Block | size_t | size_t | ACE_Message_Type | ACE_Message_Block* | char* | ACE_Allocator* | *Creates an initialized message of type <type> containing <size> bytes. The <cont> argument initializes the continuation field in the <Message_Block>. If <data> == 0 then we create and own the <data>, using <allocator> to get the data if it's non-0. If <data> != 0 we assume ownership of the <data> (and don't delete it).* | *Message_Block.cpp* |
| ~ACE_Message_Block | | | | | | | *Delete all the resources held in the message.* | *Message_Block.cpp* |
| is_data_msg | int const : O | | | | | | *Find out what type of message this is.* | *Message_Block.i* |
| msg_class | ACE_Message_Type const : O | | | | | | *Find out what class of message this is (there are two classes, <normal> messages and <high-priority> messages).* | *Message_Block.i* |
| msg_type | ACE_Message_Type const : O | | | | | | *Get type of the message.* | *Message_Block.i* |

| | | | | | | |
|---|---|---|---|---|---|---|
| **msg_type** | | ACE_ Message_ Type | | | | |
| | *Set type of the message.* | | | | ***Message_Block.i*** | |
| **set_flags** | Message_ Flags :O | Message_ Flags | | | | |
| | *Set the message flags.* | | | | ***Message_Block.i*** | |
| **clr_flags** | Message_ Flags :O | Message_ Flags | | | | |
| | *Unset the message flags.* | | | | ***Message_Block.i*** | |
| **flags** | Message_ Flags const :O | | | | | |
| | *Inspect the message flags.* | | | | ***Message_Block.i*** | |
| **msg_ priority** | u_long const :O | | | | | |
| | *Get priority of the message.* | | | | ***Message_Block.i*** | |
| **msg_ priority** | | u_long | | | | |
| | *Set priority of the message.* | | | | ***Message_Block.i*** | |
| **clone** | ACE_ Message_ Block* :O | Message_ Flags const | | | | |
| | *Return an exact "deep copy" of the message.* | | | | ***Message_Block.cpp*** | |
| **copy** | int :O | const char* | size_t | | | |
| | *Copies <n> bytes from <buf> into the Message_Block starting at the wr_ptr() offset. Return 0 if succeeds and –1 if the size of the message is too small...* | | | | ***Message_Block.cpp*** | |
| **base** | char* const :O | | | | | |
| | *Get message data.* | | | | ***Message_Block.i*** | |
| **base** | | char* | size_t | Message_Flags | | |
| | *Set message data.* | | | | ***Message_Block.i*** | |
| **end** | char* const :O | | | | | |
| | *Return a pointer to 1 past the end of the data in a message.* | | | | ***Message_Block.i*** | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **rd_ptr** | char* : O | | | | | | *Get the read pointer.* | ***Message_Block.i*** |
| **rd_ptr** | | char* | | | | | *Set the read pointer to <ptr>.* | ***Message_Block.i*** |
| **rd_ptr** | | size_t | | | | | *Set the read pointer ahead <n> bytes.* | ***Message_Block.i*** |
| **wr_ptr** | char* : O | | | | | | *Get the write pointer.* | ***Message_Block.i*** |
| **wr_ptr** | | char* | | | | | *Set the write pointer to <ptr>.* | ***Message_Block.i*** |
| **wr_ptr** | | size_t | | | | | *Set the write pointer ahead <n> bytes.* | ***Message_Block.i*** |
| **length** | size_t const : O | | | | | | *The length of a message is computed as the length between the wr_ptr() – rd_ptr ()). This is to Get the length of the message* | ***Message_Block.i*** |
| **length** | const | size_t | | | | | *The length of a message is computed as the length between the wr_ptr() – rd_ptr ()). This is to Set the length of the message* | ***Message_Block.i*** |
| **size** | size_t const : O | | | | | | *The size of the allocated buffer is the total amount of space allocated. This is to Get the total amount of space in the message.* | ***Message_Block.i*** |
| **size** | int : O | size_t | | | | | *The size of the allocated buffer is the total amount of space allocated. This is to Set the total amount of space in the message. Returns 0 if successful, else –1.* | ***Message_Block.i*** |
| **cont** | ACE_ Message_ Block* const : O | | | | | | *The continuation field is used to chain together composite messages. This is to Get the continuation field.* | ***Message_Block.i*** |

| | | | | | | |
|---|---|---|---|---|---|---|
| **cont** | | ACE_ Message_ Block* | | | | |
| | *The continuation field is used to chain together composite messages. This is to Set the continuation field.* | | | | ***Message_Block.i*** | |
| **next** | ACE_ Message_ Block* const : O | | | | | |
| | *The <next_> pointer points to the <Message_Block> directly ahead in the Message_Queue. This is to Get link to next message.* | | | | ***Message_Block.i*** | |
| **next** | | ACE_ Message_ Block* | | | | |
| | *The <next_> pointer points to the <Message_Block> directly ahead in the Message_Queue. This is to Set link to next message.* | | | | ***Message_Block.i*** | |
| **prev** | ACE_ Message_ Block* const : O | | | | | |
| | *The <prev_> pointer points to the <Message_Block> directly ahead in the Message_Queue. This is to Get link to prev message.* | | | | ***Message_Block.i*** | |
| **prev** | | ACE_ Message_ Block* | | | | |
| | *The <prev_> pointer points to the <Message_Block> directly ahead in the Message_Queue. This is to Set link to prev message.* | | | | ***Message_Block.i*** | |
| **dump** | | | | | | |
| | *Dump the state of an object.* | | | | ***Message_Block.cpp*** | |

*<u>Note#1:</u> The const after the type of the access program means that the access program is not allowed to change the private data structure of this class.*
*<u>Note#2:</u> The const before the type of the access program means that the value returned by this access program can be assigned to a constant parameter.*
*<u>Note#3:</u> : O indicates that that function returns a value, and will be listed under the VALUES section.*

#### C2.2.2.2. Inherited Public ACCESS PROGRAMS: None

## C2.2.3. CANONICAL REPRESENTATION

### C2.2.3.1. CANONICAL REPRESENTATION of This Component/Class

$S \equiv \{\langle Message\_Block\_Field_i \rangle_{i=1}^{12} \mid \langle Fields\_Types \rangle_{i=1}^{12} \} \cup \{\langle\rangle\}$

***Lexicon for S:***

$\langle Message\_Block\_Field_i \rangle_{i=1}^{12}$ ≡ <flags_, base_, cur_size_, max_size_, rd_ptr_, wr_ptr_, type_, priority_, cont_, next_, prev_, allocator_>

$\langle Fields\_Types \rangle_{i=1}^{12}$ ≡ <Message_Flags, char*, size_t, size_t, char*, char*, ACE_Message_Type, u_long, ACE_Message_Block*, ACE_Message_Block*, ACE_Message_Block*, ACE_Allocator*>

### C2.2.3.2. References To CANONICAL REPRESENTATIONs Of Inherited Components/Classes

None

## C2.2.4. EQUIVALENCES

### C2.2.4.1. EQUIVALENCES of This Component

In this section, I will specify the changes to the current state of an object (i.e. denoted by s), that are made by the different access programs in this class module.

ACE_Message_ Block ([s,],) ≡ (*s*= <"\0", "\0", "\0", "\0", "\0", "\0", #MB_NORMAL#, "\0", "\0", "\0", "\0", "\0">)

ACE_Message_Block([s,], data) ≡ (*s*= <#DONT_DELETE#, data, "\0", "\0", "\0", "\0", #MB_NORMAL#, "\0", "\0", "\0", "\0", "\0">)

ACE_Message_Block([s,], size, msg_type, msg_cont, msg_data, alloc) ≡

<table>
<tr><th colspan="2">Condition</th><th>EQUIVALENCE</th></tr>
<tr><td rowspan="2">(msg_data = 0)<br>∧</td><td>(alloc=0)</td><td>(<i>s</i> = <"\0", <i>NAlloc</i>, size, size, <i>NAlloc</i>, <i>NAlloc</i>, msg_type, "\0", msg_cont, "\0", "\0", "\0">)<br><u><i>Lexicon:</i></u><br><i>NAlloc</i> = new char[size]</td></tr>
<tr><td>(alloc≠0)</td><td>(<i>s</i> = <"\0", <i>MAllocc</i>, size, size, <i>MAllocc</i>, <i>MAllocc</i>, msg_type, "\0", msg_cont, "\0", "\0", alloc>)<br><u><i>Lexicon:</i></u><br><i>MAllocc</i> = (char *) alloc -> malloc(size)</td></tr>
<tr><td colspan="2">(msg_data ≠0)</td><td>(s = <#DONT_DELETE#, msg_data, size, size, msg_data, msg_data, msg_type, "\0", msg_cont, "\0", "\0">)</td></tr>
</table>

~ACE_Message_Block ([s,],) ≡ (s = <>)

is_data_msg([s,],) ≡ NC(s)

msg_class([s,],) ≡ NC(s)

msg_type([s,],) ≡ NC(s)

msg_type([s,], type) ≡ (s | ((s_type_field = type) ∧ NC(s_nontype_fields)))

set_flags([s,], more_flags) ≡ (s | ((s_flags_field = more_flags) ∧ NC(s_nonflags_fields)))

clr_flags([s,], less_flags) ≡ (s | ((s_flags_field = less_flags) ∧ NC(s_nonflags_fields)))

flags([s,],) ≡ NC(s)

msg_ priority([s,],) ≡ NC(s)

msg_ priority([s,], priority) ≡ ((s_Priority_field = priority) ∧ NC(s_nonpriority_fields)))

clone([s,], mask) ≡ NC(s)

copy([s,], buf, n) ≡ NC(s)

base([s,],) ≡ NC(s)

base([s,], msg_data, msg_length, msg_flags) ≡ (s=<msg_flags, msg_data, msg_length, msg_length, msg_data, msg_data, "/0", "/0", "/0", "/0", "/0", "/0">)

end([s,],) ≡ NC(s)

rd_ptr([s,],) ≡ NC(s)

rd_ptr([s,], new_ptr) ≡ (s | ((s_RdPtr_field = new_ptr) ∧ NC(s_nonRdPtr_fields)))

rd_ptr([s,], n) ≡ (s | ((s_RdPtr_field = s_RdPtr_field + n) ∧ NC(s_nonRdPtr_fields)))

wr_ptr([s,],) ≡ NC(s)

wr_ptr([s,], new_ptr) ≡ (s | ((s_WrPtr_field = new_ptr) ∧ NC(s_nonWrPtr_fields)))

wr_ptr([s,], n) ≡ (s | ((s_WrPtr_field = s_WrPtr_field + n) ∧ NC(s_nonWrPtr_fields)))

length([s,],) ≡ NC(s)

length([s,], len) ≡ (s | ((s_WrPtr_field = s_RdPtr_field + len) ∧ NC(s_nonWrPtr_fields)))

**size([s,],)** $\equiv$ *NC*(s)

**size([s,], len)** $\equiv$

| Condition | EQUIVALENCE |
|---|---|
| (len < *s_maxsize_field*) | ((*s_CurSize_field* = len) $\wedge$ *NC*(*s_nonCurSize_fields*)) |
| (len $\geq$ *s_maxsize_field*) | (s =<*NC*($Fld_1$), nBase, nCurSize, nMaxSize, nRdPtr, nWrPtr, *NC*($Fld_7$), *NC*($Fld_8$), *NC*($Fld_9$), *NC*($Fld_{10}$), *NC*($Fld_{11}$), *NC*($Fld_{12}$)>) |

***Private Lexicon:***

**nBase** = a new address randomly allocated by the system
**nCurSize** = **nMaxSize** = **len**
**nRdPtr**= **nBase + rd_ptr_ – base_**
**nWrPtr** = **nBase + wr_ptr_ – base_**
**$Fld_i$** = **$Message\_Block\_Field_i$**

**cont([s,],)** $\equiv$ *NC*(s)

**cont([s,],next_block)** $\equiv$ (s | ((*s_Cont_field* = next_block) $\wedge$ *NC*(*s_nonCont_fields*)))

**next([s,],)** $\equiv$ *NC*(s)

**next([s,],next_block)** $\equiv$ (s | ((*s_Next_field* = next_block) $\wedge$ *NC*(*s_nonNext_fields*)))

**prev([s,],)** $\equiv$ *NC*(s)

**prev([s,],next_block)** $\equiv$ (s | ((*s_Prev_field* = next_block) $\wedge$ *NC*(*s_nonPrev_fields*)))

**dump([s,],)** $\equiv$ (s=<'\0", '\0", '\0", '\0", '\0", '\0", '\0", '\0", '\0", '\0", '\0", '\0">)

### C2.2.4.2. References To EQUIVALENCES Of Inherited Components

None

## C2.2.5. VALUES

### C2.2.5.1. VALUES of This Component

| | |
|---|---|
| *o*(is_data_ msg([s,],)) = | x, x $\in$ {#MB_DATA#, #MB_PROTO#, #MB_PCPROTO#} |
| *o*(msg_class([s,],)) = | x, x $\in$ {#MB_PRIORITY#, #MB_NORMAL#} |
| *o*(msg_type([s,],)) = | x, x $\in$ ACE_Message_Type enum |
| *o*(set_flags([s,], more_flags)) = | *VAL(s_flags_field)*, *VAL( s_flags_field)* = more_flags |
| *o*(clr_flags([s,], less_flags)) = | *VAL(s_flags_field)* , *VAL( s_flags_field)* =less_flags |

| | |
|---|---|
| $o$(**flags([s,],)**) = | ***VAL(s_flags_field)*** |
| $o$(**msg_ priority([s,],)**) = | ***VAL(s_priority_field)*** |
| $o$(**clone([s,], mask)**) = | [s, <**Message_Block_Field**$_i$> i=1..12] // return exact deep copy of the object |
| $o$(**copy([s,], buf, n)**) = | x, x ∈ {0, –1} // if succeeds, else –1 |
| $o$(**base([s,],)**) = | ***VAL(s_base_field)*** |
| $o$(**end([s,],)**) = | ***VAL(s_base_field) + VAL( s_MaxSize_field)*** |
| $o$(**rd_ptr([s,],)**) = | ***VAL(s_RdPtr_field)*** |
| $o$(**wr_ptr([s,],)**) = | ***VAL(s_WrPtr_field)*** |
| $o$(**length([s,],)**) = | ***VAL(s_WrPtr_field) – VAL(s_RdPtr_field)*** |
| $o$(**size([s,],)**) = | ***VAL(s_CurSize_field)*** |
| $o$(**size([s,], len)**) = | x, x ∈ {0, –1} // 0 if succeeds, else –1 |
| $o$(**cont([s,],)**) = | ***VAL(s_Cont_field)*** |
| $o$(**next([s,],)**) = | ***VAL(s_Next_field)*** |
| $o$(**prev([s,],)**) = | ***VAL(s_Prev_field)*** |

### C2.2.5.2. References To VALUES Of Inherited ACCESS Programs

None

## C2.2.6. LEXICON

***ERROR***(**%Error Message%**) ≡ An auxiliary function that displays the **%Error Message%**.

***NC***(**x**) ≡ **(x' = 'x)**

***VAL***(**x**) ≡ An auxiliary function that returns the value stored in the variable **x**

***NumFields*** ≡ 12

| *Object Fields Order* | | | *Object Fields* | | |
|---|---|---|---|---|---|
| *FlagsFieldOrder* | ≡ | 1 | *s_flags_field* | ≡ | *Message_Block_Field*$_{FlagsFieldOrder}$ |
| *BaseFieldOrder* | ≡ | 2 | *s_base_field* | ≡ | *Message_Block_Field*$_{BaseFieldOrder}$ |
| *CurSizeFieldOrder* | ≡ | 3 | *s_CurSize_field* | ≡ | *Message_Block_Field*$_{CurSizeFieldOrder}$ |
| *MaxSizeFieldOrder* | ≡ | 4 | *s_MaxSize_field* | ≡ | *Message_Block_Field*$_{MaxSizeFieldOrder}$ |

| | | | | | |
|---|---|---|---|---|---|
| *RdPtrFieldOrder* | ≡ | 5 | *s_RdPtr_field* | ≡ | $Message\_Block\_Field_{RdPtrFieldOrder}$ |
| *WrPtrFieldOrder* | ≡ | 6 | *s_WrPtr_field* | ≡ | $Message\_Block\_Field_{WrPtrFieldOrder}$ |
| *TypeFieldOrder* | ≡ | 7 | *s_type_field* | ≡ | $Message\_Block\_Field_{TypeFieldOrder}$ |
| *PriorityFieldOrder* | ≡ | 8 | *s_priority_field* | ≡ | $Message\_Block\_Field_{PriorityFieldOrder}$ |
| *ContFieldOrder* | ≡ | 9 | *s_Cont_field* | ≡ | $Message\_Block\_Field_{ContFieldOrder}$ |
| *NextFieldOrder* | ≡ | 10 | *s_Next_field* | ≡ | $Message\_Block\_Field_{NextFieldOrder}$ |
| *PrevFieldOrder* | ≡ | 11 | *s_Prev_field* | ≡ | $Message\_Block\_Field_{PrevFieldOrder}$ |
| *AllocatorFieldOrder* | ≡ | 12 | *s_Allocator_field* | ≡ | $Message\_Block\_Field_{AllocatorFieldOrder}$ |

| *Predicates* | | *Equivalences* |
|---|---|---|
| *s_nonflags_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne FlagsFieldOrder))$ |
| *s_nonbase_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne BaseFieldOrder))$ |
| *s_nonCurSize_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne CurSizeFieldOrder))$ |
| *s_nonMaxSize_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne MaxSizeFieldOrder))$ |
| *s_nonRdPtr_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne RdPtrFieldOrder))$ |
| *s_nonWrPtr_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne WrPtrFieldOrder))$ |
| *s_nontype_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne TypeFieldOrder))$ |
| *s_nonpriority_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne PriorityFieldOrder))$ |
| *s_nonCont_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne ContFieldOrder))$ |
| *s_nonNext_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne NextFieldOrder))$ |
| *s_nonPrev_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne PrevFieldOrder))$ |
| *s_nonAllocator_fields* | ≡ | $Message\_Block\_Field_i, ((1 \le i \le NumFields) \wedge (i \ne AllocatorFieldOrder))$ |

# Appendix D

# ACRONYMS, VARIABLES, SYMBOLS, CONSTANTS, AND AUXILIARY FUNCTIONS

## D1.1. Acronyms

| | |
|---|---|
| API: | Application Programming Interface |
| CGI: | Component Guide Document |
| CISD: | Component Interface Specifications Document |
| CIDSD: | Component Internal Design Specifications Document. |
| CO: | Component–Oriented |
| ***Fr***: | Framework. |
| IPC: | Inter–Process Communication |
| OO: | Object–Oriented |
| OS: | Operating Systems |
| ***PA***: | Path A. |
| ***PB***: | Path B. |
| ***SP***: | Shared–Pool. |
| TRIO: | Telecommunication Research Institute of Ontario |
| TTS: | Table Tool System |

## D1.2. Variables

$\mathit{CANONICAL\_REPRESENTATION\_OfComponent}_{ij}$: Represents the canonical representation for the $ij^{th}$ component.

*CANONICAL_REPRESENTATION_OfInternalClass$_{ijl}$*: Represents the canonical representation for the $l^{th}$ internal class of the $ij^{th}$ component.

*CGD*: Represents the **Component Guide Document** as described in section 6.2.

*CGD_DescriptiveOverview*: Represents the first chapter of the proposed CGD document. This chapter should describe the various aspects of the CO–Toolkit under consideration.

*CGD_PictorialOverview*: Represents the second chapter of the proposed CGD document. This chapter should provide a set of pictorial views that describe several things such as the file structure used for the considered CO–Toolkit, the various relations across the frameworks and among their components (i.e. such as inheritance and association relations), and etc.

*CGD_IndexedQuickReferenceTables*: Represents the third chapter of the proposed CGD document. This chapter should provide a set of index table that can help users as well as maintainers to find out what service(s) is (are) provided by each component in the CO–Toolkit under consideration.

*CIDSD*: Represents the Component Internal Design Specifications Document as described in section 6.4.

*CIDSD_Chapter_Framework$_i$*: Represents the CIDSD chapter that provides the internal design specifications for the $i^{th}$ framework.

*CIDSD_Chapter_SharedPool$_k$*: Represents the CIDSD chapter that provides the internal design specifications for the $k^{th}$ shared pool.

*CIDSD_Section_Component$_{ij}$*: Represents the section dedicated for the internal design specifications of $ij^{th}$ component as a part of the CIDSD chapter for the $i^{th}$ framework.

*CIDSD_Section_SharedClass$_{kj}$*: Represents the section dedicated for the internal design specifications of the $kj^{th}$ shared class as a part of the CIDSD chapter for the $k^{th}$ shared–pool.

*CIDSD_Section_OtherSharedEntity$_{kj}$*: Represents the section dedicated for the internal design of the $kj^{th}$shared non–class entities as a part of the CIDSD chapter for the $k^{th}$ shared–pool.

*CIDSD_SubSection_Main$_{ij}$*: Represents the sub–section that provides the internal design specifications of the main part of the $ij^{th}$ component.

*CIDSD_SubSection_InternalClass$_{ijl}$*: Represents the sub–section that provides the internal design specifications for the $l^{th}$ internal class in the $ij^{th}$ component.

*CIDSD_SubSection_SharedClass_Public$_{kj}$*: Represents the sub–section that provides the internal design specifications of the public section of the $kj^{th}$ shared class.

*CIDSD_SubSection_SharedClass_Protected$_{kj}$*: Represents the sub–section that provides the internal design specifications of the protected section of the $kj^{th}$ shared class.

*CIDSD_SubSection_SharedClass_Private$_{kj}$*: as explained in (6–21), this element represents the internal design specifications of the private section of the $kj^{th}$ shared class.

*CIDSD_SubSection_Main_DISPLAY$_{ijd}$*: Represents the $d^{th}$ DISPLAY of those used to specify the internal design of the Main part of the $ij^{th}$ component.

*CIDSD_SubSection_InternalClass_Public$_{ijl}$*: Represents the sub–section that provides the internal design specifications of the public section of the $l^{th}$ internal class of the $ij^{th}$ component.

*CIDSD_SubSection_InternalClass_Protected$_{ijl}$*: Represents the sub–section that provides the internal design specifications of the protected section of the $l^{th}$ internal class of the $ij^{th}$ component.

*CIDSD_SubSection_InternalClass_Private$_{ijl}$*: Represents the sub–section that provides the internal design specifications of the private section of the $l^{th}$ internal class of the $ij^{th}$ component.

*CISD*: Represents the **Component Interface Specifications Document** as described in section 6.3.

*CISD_Chapter_Framework$_i$*: Represents the CISD chapter that provides the interface specifications for the $i^{th}$ framework.

*CISD_Section_Component$_{ij}$*: Represents the section dedicated for the interface specifications of $ij^{th}$ component as a part of the CISD chapter for the $i^{th}$ framework.

*CISD_SubSection_CHARACTERISTICS$_{ij}$*: Represents the general characteristics of the $ij^{th}$ component.

*CISD_SubSection_SYNTAX$_{ij}$*: Represents the signatures of the access programs of the $ij^{th}$ component.

*CISD_SubSection_CANONICAL_REPRESENTATION$_{ij}$*: Represents the canonical representation for the $ij^{th}$ component.

*CISD_SubSection_EQUIVALENCES$_{ij}$*: Represents the equivalences for the $ij^{th}$ component.

*CISD_SubSection_VALUES$_{ij}$*: Represents the values returned by the access programs of the $ij^{th}$ component.

*CISD_SubSection_LEXICON$_{ij}$*: Represents the lexicon for the interface specifications of the $ij^{th}$ component.

*C_Files$_{ij}$*: Represents the set of files that are used to implement $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

*COToolkit*: Represents any Component–Oriented Toolkit that can provide reusable components to be used to build new applications.

*CO_Files*: Represents the overall set of files that are used to implement the CO_toolkit that is under consideration.

***Component***$_{ij}$: Represents the $j^{th}$ component in the $i^{th}$ framework.

***EQUIVALENCES_OfComponent***$_{ij}$: Represents the set of equivalent states to the canonical one. An object of the $ij^{th}$ component goes into certain one of these equivalent states whenever a certain access program is invoked on that object.

***Framework***$_{i}$: Represents the $i^{th}$ framework.

***Fr_Files***$_{i}$: Represents the set of files that are used to implement $i^{th}$ framework of the CO_toolkit that is under consideration.

***Implementation***$_{ij}$: Represents the implementation of the $ij^{th}$ component.

***InheritedInterface***$_{ij}$: Represents the inherited interface of the $ij^{th}$ component. This refers to the interface of the methods that are inherited by that component.

***InheritedPublic_ACCESS_ProgramsByComponent***$_{ij}$: Represents the set of access programs (methods) that are publicly inherited by the $ij^{th}$ component from other components.

***intermediateFile***$_{ij}$: Represents the .h file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

***Interface***$_{ij}$: Represents the interface of the $ij^{th}$ component.

***InternalClass***$_{ijl}$: Represents the implementation of the $l^{th}$ internal class in the $ij^{th}$component.

***InternalClassesSet***$_{ij}$: Represents the set of the internal classes in the $ij^{th}$component.

***InternalClass_PublicMethod_DISPLAY***$_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ public method in the $l^{th}$ internal class of the $ij^{th}$ component.

***InternalClass_ProtectedMethod_DISPLAY***$_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ protected method in the $l^{th}$ internal class of the $ij^{th}$ component.

***InternalClass_PrivateMethod_DISPLAY***$_{ijlmd}$: Represents the $d^{th}$ Parnas DISPLAY of those required to specify the internal design of the $m^{th}$ private method in the $l^{th}$ internal class of the $ij^{th}$ component.

***InternalClass_DataStructureSpecifications***$_{ijl}$: Represents the data structure implemented in the $l^{th}$ class of the $ij^{th}$ component.

***internalFile***$_{ij}$: Represents the .i file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

***LocalInterface***$_{ij}$: Represents the local interface of the $ij^{th}$ component. This refers to the interface of the local non–inherited methods that belong to that component.

***LocalPublicMethodSignature***$_{ijlk}$: Represents the signature (interface description) of the $k^{th}$ local method in the $l^{th}$ class that belongs to the $ij^{th}$ component.

***$MainPart_{ij}$***: Represents the implementation of the main part of the $ij^{th}$ component. The main part of a component contains the component's execution entry point (i.e. main function) and may contain other functions that do not belong to the internal classes of that component.

***$OtherSharedEntitiesSet_k$***: Represents the set of other shared entities in the $k^{th}$ shared–pool.

***$OtherSharedEntity_{kf}$***: Represents the $f^{th}$ other shared entity in the $k^{th}$ shared–pool.

***$PrivateMethod_{ijlf}$***: Represents the implementation of the $f^{th}$ private method of he $l^{th}$ internal class in the $ij^{th}$component.

***$PrivateObject_{ijlk}$***: Represents the $k^{th}$ private object of the $l^{th}$ internal class in the $ij^{th}$ component.

***$PrivateSection_{ijl}$***: Represents the implementation of the private section of the $l^{th}$ internal class of the $ij^{th}$component.

***$ProtectedMethod_{ijlf}$***: Represents the implementation of the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$component.

***$ProtectedObject_{ijlk}$***: Represents the $k^{th}$ protected object of the $l^{th}$ internal class of the $ij^{th}$ component.

***$ProtectedSection_{ijl}$***: Represents the implementation of the protected section of the $l^{th}$ internal class in the $ij^{th}$component.

***$PublicConstantParameters_{ij}$***: Represents the set of public constant that appear in the interface specifications of the $ij^{th}$ component.

***$PublicMethod_{ijlk}$***: Represents the implementation of the $k^{th}$ public method of the $l^{th}$ internal class in the $ij^{th}$component.

***$PubliclyInheritedMethodSignature_{ijlf}$***: Represents the signature (interface description) of the $f^{th}$ method that is inherited by the $l^{th}$ class of the $ij^{th}$ component.

***$PublicPrimitiveSystemDataTypes_{ij}$***: Represents the set of primitive data types (i.e. int, boolean etc.) that appear in the interface specifications of the $ij^{th}$ component.

***$PublicSection_{ijl}$***: Represents the implementation of the public section of the $l^{th}$ internal class in the $ij^{th}$component.

***$PublicSpecificationTypes_{ij}$***: Represents the set of types that appear in the interface specifications of the $ij^{th}$ component.

***$PublicSpecificationVariables_{ij}$***: Represents the set of variables that appear in the interface specifications of the $ij^{th}$ component.

***$Public\_ACCESS\_ProgramsInComponent_{ij}$***: Represents the set of public access programs (methods) that are contained in the $ij^{th}$ component.

$\boldsymbol{P\_INs_{ijlk}}$: Represents the sequence of inputs to the $k^{th}$ public local method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{P\_OUTs_{ijlk}}$: Represents the sequence of outputs from the $k^{th}$ public local method in the $l^{th}$ internal class of the $ij^{th}$ component.

$\boldsymbol{P\_ReturnedValue_{ijlk}}$: Represents the returned value by the $k^{th}$ public local method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{P_{Inhr}\_INs_{ijlf}}$: Represents the sequence of inputs to the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{P_{Inhr}\_OUTs_{ijlf}}$: Represents the sequence of outputs from the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{P_{Inhr}\_ReturnedValue_{ijlf}}$: Represents the returned value by the $f^{th}$ publicly inherited method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{ReferencesTo\_CANONICAL\_REPRESENTATIONS\_OFInheritedComponents_{ij}}$: Represents a set of references to the canonical representations of the other components that are inherited from by the $ij^{*}$ component.

$\boldsymbol{ReferencesTo\_EQUIVALENCES\_OFInheritedComponents_{ij}}$: Represents a set of references to the equivalences of the other components that are inherited from by the $ij^{*}$ component.

$\boldsymbol{ReferencesTo\_VALUES\_OFInheritedComponents_{ij}}$: Represents a set of references to the values returned by the public access programs (methods) of the other components that are inherited from by the $ij^{*}$ component.

$\boldsymbol{ReferencesTo\_DISPLAYs\_OFPublicMethodsInheritedByInternalClass_{ijl}}$: Represents a set of references to the set of Parnas DISPLAYs that provide the internal specifications for those internal class methods that are inherited by the $l^{*}$ internal class of the $ij^{*}$ component.

$\boldsymbol{R\_INs_{ijlf}}$: Represents the sequence of inputs to the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{R\_OUTs_{ijlf}}$: Represents the sequence of outputs from the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{R\_ReturnedValue_{ijlf}}$: Represents the returned value by the $f^{th}$ protected method of the $l^{th}$ internal class in the $ij^{th}$ component.

$\boldsymbol{Sh\_Files_k}$: Represents the set of files that are used to implement $k^{th}$ shared–pool of the CO_toolkit that is under consideration.

$Sh\_File_{kl}$: Represents the $l^{th}$ file in the $k^{th}$ shared–pool.

$topFile_{ij}$: Represents the .cpp file of the $j^{th}$ component in the $i^{th}$ framework of the CO_toolkit that is under consideration.

$SharedPool_k$: Represents the $k^{th}$ shared–pool.

$SharedClass_{kj}$: Represents the $j^{th}$ class in the $k^{th}$ shared–pool.

$SOF_X(A,B)$: Represents the implementation of method X.

$TypeSpecified_{ij}$: Represents the specification type of the $ij^{th}$ component.

$VALUES\_OfComponent_{ij}$: Represents the set of values that are returned by the public access programs (methods) of the $ij^{th}$ component.

$V\_INs_{ijlf}$: Represents the sequence of inputs to the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

$V\_OUTs_{ijlf}$: Represents the sequence of outputs from the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

$V\_ReturnedValue_{ijlf}$: Represents the returned value by the $f^{th}$ private method of the $l^{th}$ internal class in the $ij^{th}$ component.

## D1.3. Symbols And Constants

$F$: Number of Frameworks in the given CO_toolkit.

$C_i$: Number of Components in the $i^{th}$ Framework.

$c_{ij}$: Represents the number of local classes used to construct the $ij^{th}$ component.

$c_k^s$: Represents the number of shared classes in the $k^{th}$ shared–pool.

$e_k^s$: Represents the number of other shared entities in the $k^{th}$ shared–pool.

$M_{ijl}^P$: Represents the number of local public methods in $l^{th}$ class of the $ij^{th}$ component.

$M_{ijl}^{P_{Inhr}}$: Represents the number of public methods inherited by the $l^{th}$ class of the $ij^{th}$ component.

$M_{ijl}^R$: Represents the number of local protected methods in the $l^{th}$ class of the $ij^{th}$ component.

$M_{ijl}^V$: Represents the number of local private methods in the $l^{th}$ class of the $ij^{th}$ component.

$NumberOfDISPLAYsForMainPartOfComponent_{ij}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the main part of the $ij^{th}$ component.

$NumberOfDISPLAYsForPublicMethod_{ijlm}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ public method in the $l^{th}$ internal class of the $ij^{th}$ component.

$NumberOfDISPLAYsForProtectedMethod_{ijlm}$: Represents the number of Parnas DISPLAYs that are required to specify the internal design of the $m^{th}$ protected method in the $l^{th}$ internal class of the $ij^{th}$ component.

$NumberOfFilesInCO\_Files$: Represents the total number of files used to implement the given CO–Toolkit.

$O^{R}_{ijl}$: Represents the number of objects in the protected section of the $l^{th}$ class of the $ij^{th}$ component.

$O^{V}_{ijl}$: Represents the number of objects in the private section of the $l^{th}$ class of the $ij^{th}$ component.

$pm^{IN}_{ijlk}$: Represents the number of inputs of the $k^{th}$ local public method in the $l^{th}$ class of the $ij^{th}$ component.

$pm^{OUT}_{ijlk}$: Represents the number of outputs of the $k^{th}$ local public method in the $l^{th}$class of the $ij^{th}$ component.

$pm^{IN_{Inhr}}_{ijlf}$: Represents the number of inputs of the $f^{th}$ public method that is inherited by the $l^{th}$ class of the $ij^{th}$ component.

$pm^{OUT_{Inhr}}_{ijlf}$: Represents the number of outputs of the $f^{th}$ public method that is inherited by the $l^{th}$ class of the $ij^{th}$component.

$rm^{IN}_{ijlf}$: Represents the number of inputs of the $f^{th}$ local protected method in the $l^{th}$ class of the $ij^{th}$component.

$rm^{OUT}_{ijlf}$: Represents the number of outputs of the $f^{th}$ local protected method in the $l^{th}$class of the $ij^{th}$component.

$S$: Number of Shared–Pools.

$s_k$: Number of Files in the $k^{th}$ Shared–Pool.

$vm^{IN}_{ijlf}$: Represents the number of inputs of the $f^{th}$ local private method in the $l^{th}$ class of the $ij^{th}$component.

$vm^{OUT}_{ijlf}$: Represents the number of outputs of the $f^{th}$ local private method in the $l^{th}$class of the $ij^{th}$ component.

## D1.4. Auxiliary Functions

$Associates(X, Y) \equiv \begin{pmatrix} UniDirectionalAssociation(X, Y) \vee \\ BiDirectionalAssociation(X, Y) \end{pmatrix}$: An auxiliary function that returns TRUE if classes X and Y associate with each other.

$BiDirectionalAssociation(X, Y) \equiv \begin{bmatrix} One\_to\_One\_BiDirectionalAssociation(X, Y) \vee \\ One\_to\_Many\_BiDirectionalAssociation(X, Y) \vee \\ Many\_to\_Many\_BiDirectionalAssociation(X, Y) \end{bmatrix}$: An auxiliary function that returns TRUE if X and Y have a form of bi–directional association with each other.

$CARD(FLAG, X) \equiv$ This is an auxiliary function that returns the number of elements in the set X. For convenience and as shown in the next table, the *FLAG* is used to indicate the type of elements in the set X.

*Category*(*X*): An auxiliary function that returns the category of the file X. It could be 'top', 'intermediate', 'internal', or 'shared'.

*FileName*(*X*): An auxiliary function that returns the filename of file X.

$Fr\_Name(Fr\_Files_i)$: An auxiliary function that returns the name of the framework that is implemented using the $Fr\_Files_i$ file subset.

*Frs_PermittedToAccessSP*(*k*): An auxiliary function that returns the set of frameworks that are permitted to access the $k^{th}$ shared–pool.

*Includes*(*X*, *Y*): An auxiliary Boolean function that returns TRUE or FALSE. It returns TRUE when file X is "#include"ing file Y, otherwise it returns FALSE.

*InheritanceCategory*(*Inherits*(*X*, *Y*)): An auxiliary function that returns the inheritance category when component X inherits from component Y. The returned value is either "Public", "Protected", or "Private".

*Inherits*(*X*, *Y*): An auxiliary boolean function that returns TRUE if component X does inherit from component Y. Otherwise, it returns FALSE.

$LENGTH(FLAG, Y) \equiv$ This function returns the number of elements in the sequence Y. For convenience and as shown in the next table, the FLAG is used to indicate the type of elements in the sequence Y.

*Links*(*X*, *Y*): An auxiliary Boolean function that returns TRUE or FALSE. It returns TRUE when file X contains parts of file Y, otherwise, it returns FALSE.

$Many\_to\_Many\_BiDirectionalAssociation(X, Y) \equiv$ An auxiliary function that returns TRUE if Classes X and Y each contains multiple objects of the other.

$One\_to\_One\_BiDirectionalAssociation(X, Y) \equiv$ An auxiliary function that returns TRUE if Classes X and Y each contains an object of the other.

*One_to_One_UniDirectionalAssociation(X, Y)* ≡ An auxiliary function that returns TRUE if Class X contains an object of Class Y while Y does not contain any object of X.

*One_to_Many_BiDirectionalAssociation(X, Y)* ≡ An auxiliary function that returns TRUE if Class X contains a set of objects of Class Y while Y contains only one object of class X.

*One_to_Many_UniDirectionalAssociation(X, Y)* ≡ An auxiliary function that returns TRUE if Class X contains a set of objects of Class Y while Y does not contain any object of X.

*ProvidesInterfaceSpecifications(X, Y)* ≡ This is an auxiliary predicate that returns TRUE if the interface specifications of the Sub–System Y are provided in the portion X of the CISD document.

*ProvidesDesignSpecifications(X, Y)* ≡ This is an auxiliary predicate that returns TRUE if the design and implementation specifications of the Sub–System Y are provided in the portion X of the CIDSD document.

*SEQUENCE*$[f_1, f_2, \ldots, f_n]$: is used to denote a sequence of file names as per Hoffman's notation [25].

*Type(X)*: An auxiliary function that returns the data type of the argument *X*.

$UniDirectionalAssociation(X, Y) \equiv \begin{pmatrix} One\_to\_One\_UniDirectionalAssociation(X, Y) \vee \\ One\_to\_Many\_UniDirectionalAssociation(X, Y) \end{pmatrix}$: An auxiliary function that returns TRUE if X and Y have a form of uni–directional association with each other.

*UsedToImplement(X, Y)* ≡ This is an auxiliary predicate that returns TRUE if the Source Code of the Sub–System Y is provided in the Set of Files X.

# REFERENCES


[1] Abrahm R., et al, "Table Tool System User's Guide" Revision 1.1, Software Engineering Group, Communication Research Laboratory, McMaster University, Hamilton, Ontario, 1996.

[2] Alencar, P.S.C., Cowan, D.D., Nelson, T., Nova, L., Radaideh, M., and Schauer R., "CASCON'97 Workshop Report: Patterns and Frameworks", IBM Toronto Technical Report TR–74.161, CASCON'97 Workshop Reports, IBM Toronto Laboratory, April 1998.

[3] Ayers, K. E., "The MVC paradigm in smalltak4", Dr. Bobb's Journal – Software Tools for the professional Programmer, 15(7): pp 168–175. November 1990.

[4] Bauer, B., "Documenting Complicated Programs", CRL Report No. 316, Communication Research Laboratory, Department of Electrical and Computer Engineering, McMaster University, 1995.

[5] Black, A., et al, "Object–Structure in the Emerald System", OOPSLA'86 Conference Proceedings, SIGPLAN Notices (Special Issue), Vol. 21, n0. 11, pp 78–86. 1986.

[6] Booch, G., "Object–Oriented Analysis and Design With Applications", 2nd Edition. Redwood City, California: Benjamin/Cummings, 1993.

[7] Briton, K.H., Parnas, D.L., "A–7E Software Module Guide", NRL Memorandum Report 4702, Naval Research Laboratory, Computer Science Branch, Information Technology Division, Washington, D.C., December 8, 1981.

[8] Bruno Lague, Bell Canada Presentation on the DATRIX ToolKit, CRL–B102, McMaster University, Dec. 5 1996. A user manual for DATRIX is available as well.

[9] Bruno Lague, Bell Canada, Meeting and Discussions, January, 1997.

[10] BYTE, "ComponentWare – Object–oriented computing has failed. But component software, such as Visual Basic's custom controls, is succeeding. Here's why", BYTE Magazine, May 1994, pp46–56.

[11] Candenhead, R., "Teach Yourself Java 1.1 Programming in 24 Hours", 1st edition, Sam.net Publishing, 1997.

[12] Cardelli, L. and Wegner, P., "On Understanding Types, Data Abstraction, and Polymorphism", ACM Computing Surveys vol. 17 (4), 1985.

[13] Coplien, J.O., "Advanced C++ Programming Styles and Idioms", Addison Wesley, 1992.

[14] Ciupke, Oliver, "Automatic Detection of Design Problems in Object–Oriented Reengineering", FZI Forschungszentrum Informatik, 1999. This can be obtained from the author at e–mail: ciupke@fzi.de

[15] Cox, B.J., "Object–Oriented Programming, An Evolutionary Approach". Addison Wesley, 1991.

[16] Eckel, B., "Thinking In JAVA", MindView Inc., 1997.

[17] Embley, D. W., et al, "Object–Oriented Systems Analysis, A Model–Driven Approach", Prentice–Hall, Englewood Cliffs, New Jersey, 1992.

[18] Fayad, M. E., "Object–Oriented Software Engineering: Problems and Perspectives", Ph.D. thesis, University of Minnesota, 1994.

[19] Fayad, M., and Shmidt, D., "Object–Oriented Application Frameworks", guest editorial, ACM Communications, Special Issue on Object–Oriented Application Frameworks, Vol. 40, No. 10, October 1997.

[20] Garfinkel, S.L. and Mahoney, M., "NEXTSTEP PROGRAMMING STEP ONE: Object–Oriented Applications", Springer–Verlag, 1993.

[21] Ghezzi, C., Jazayeri, M., and Mandrioli, D., "Fundamentals of Software Engineering", Prentice–Hall, 1991.

[22] Goldberg, A., "Information models, views, and controllers", Dr. Dobb's Journal: Software Tools for the Professional Programmer, 15(7): pp 54–61, July 1990.

[23] Halmos, P. R., "Naive Set Theory", Springer–Verlag, 1974.

[24] Hathaway, B., "Object–Orientation FAQ", Geodesic Systems, Inc., Cyberdyne Systems Corporation, 1995.

[25] Hoffman, D., and Strooper, P., "Software Design, Automated Testing, and Maintenance, A PRACTICAL APPROACH", International Thomson Computer Press, 1995.

[26] Ladd, S.R., "C++ Techniques and Applications", M&T Books, 1990.

[27] Lakoff, G., "Women, Fire, and Dangerous Things: What Categories Reveal About The Mind", UOC Press, 1987.

[28] Lakos, J., "Large–Scale C++ Software Design", ISBN 0–201–63362–0, Addison–Wesley, 1996

[29] McGraw–Hill Edition, "Random House Webster's College Dictionary", Random House, Inc., 1991.

[30] Meyer, B., "Object–Oriented Software Construction", , Prentice Hall, 1988.

[31] Meyer, B., "On To Components", Computer Magazine, pp139–143, January 1999.

[32] Mikhajlov, L., and Sekerinski, E., "The Fragile Base Class Problem and Its Solution", Turku Center for computer science, TUCS Technicla Report No 117. June 1997.

[33] Mugridge, W.B., et al., "Multi–Methods in a Statically–Typed Programming Language", Proc. ECOOP, 1991.

[34] Object International Inc., "Together/C++ User Manual", obtained from the Together/C++ CD, 1997.

[35] Ousterhout, J. K., "Tcl and the Tk Toolkit", Addison–Wesley, April 1994, ISBN 0–201–6337–X

[36] Parnas D.L., "On The Criteria To Be Used In Decomposing Systems Into Modules", Communications Of The ACM, Vol. 15, Number 12, December 1972.

[37] Parnas, D.L., "On The Design and Development Of Program Families", IEEE Transactions On Software Engineering, Volume SE–2, Number 1, March 1976.

[38] Parnas, D.L., "Designing Software For Ease of Extension and Contraction", Proceedings of the 3rd International Conference on Software Engineering (10–12 May 1978), pp. 264–277. Later published in IEEE Transactions on Software Engineering, March 1979, pp. 128–138.

[39] Parnas D.L., and Mady J. "Functional Documentation for Computer Systems Engineering (version 2)", CRL Report 237, McMaster University, CRL, TRIO, Hamilton, Ontario, Canada, September 1991. Published in the Science of Computer Programming 25 (1995) 41–61.

[40] Parnas D.L., Mady J., Igleweski M., "Formal Documentation of Well–Structured Programs", CRL Report 259, McMaster University, CRL, TRIO, Hamilton, Ontario, Canada, September 1992.

[41] Radaideh, M.A., "Towards Documenting, Designing and Organizing Toolkits That Are Composed of Sets of Object–Oriented Components", Proc. TRIO–RETREAT, Kingston, Ontario, Canada, May 6–8, 1997.

[42] Rumbaugh, J., "Modeling modes and viewing views: A look at the model–view controller framework", Journal of Object–Oriented Programming, pp. 14–, May 1994.

[43] Schmidt, D.C., " The ADAPTIVE Communication Environment, An Object–Oriented Network Programming ToolKit for Developing Communication Software", Department of Computer Science,Washington University, St. Louis, MO 63130.

[44] Schmidt, D.C., "The Reactor: An Object–Oriented Interface for Event–Driven UNIX I/O Multiplexing (part 1 of 2)," C++ Report, vol. 5, February 1993.

[45] Schmidt, D.C., "The Object–Oriented Design and Implementation of the Reactor: A C++ Wrapper for UNIX I/O Multiplexing (part 2 of 2)," C++ Report, vol. 5, September 1993.

[46] Schmidt, D.C., "Reactor: An Object Behavioral Pattern for concurrent Event Demultiplexing and Event Handler Dispatching," Pattern Languages of Program Design (J.O. Coplienand D.C. Schmidt, eds.) Reading, MA: Addison–Wesley, 1995.

[47] Schmidt, D.C., "ASX: an Object–Oriented Framework for Developing Distributed Applications" Proceeding of the 6th USENIX C++ Technical Conference, (Cambridge, Massachusetts), USENIX Association, April 1994.

[48] Schmidt, D.C., "IPC_SAP: An Object–Oriented Interface to Interprocess Communication Services", C++ Report, vol. 4, November/December 1992.

[49] Schmidt, D.C. "Acceptor and Connector: Design Patterns for Active and Passive Establishment of Network Connections", Workshop on Pattern Languages of Object–Oriented Programs at ECOOP '95, (Aarhus, Denmark), August 1995.

[50] Schmidt, D.C., Harrison, T., and Al–Shaer, E., "Object–Oriented Components for High–Speed Network Programming", proceeding of the conference on Object–Oriented Technologies, (Monterey, CA), USENIX, June 1995.

[51] Schmidt, D.C., "An OO Encapsulation of Lightweight OS Concurrency Mechanisms in the ACE Toolkit" Tech. Rep. WUCS–95–31, Washington University, St. Louis, Sept. 1995.

[52] Schmidt, D.C, and Suda, T., "An Object–Oriented Framework for Dynamically Configuring Extensible Distributed Communication Systems", IEE/BCS Distributed Systems Engineering Journal (Special Issue on Configurable Distributed Systems), vol. 2, pp. 280–293, December 1994.

[53] SERG: The Software Engineering Research Group, Communication Research Laboratory, Department of Electrical and Computer Engineering, Mcmaster University. Under the supervision of Professor David L. Parnas.

[54] Shen, H., and Parnas D.L , a project on formal specifications for OO–programs. A project for Bell Canada under the supervision of Dr. Parnas.

[55] Szyperski, Clemens, "Component–Oriented Programming: A Refined Variation on Object–Oriented Programming", School of Computing Science, Queensland University, Brisbane, Australia, The Oberon Tribune, Vol 1, No 2, December 1995.

[56] Szyperski, Norbert, "Component Software – A Market on the Verge of Success", University of Cologne, Germany, The Oberon Tribune, Vol 2, No 1, 1996.

[57] Wong, K., "Rigi User's Manual Version 5.4.3", Nov. 1, 1996. This manual is provided by the Rigi software system.

## Other Related References

[58] Agrawal, R., et al., "Static Type Checking of Multi–Methods", OOPSLA 91, Object–Oriented Programming Systems, Languages, and Applications. ACM Press. Addison Wesley, 1991.

[59] Aho, A.V., Sethi, R., and Ullman, J.D., "Compilers: Principles, Techniques, and Tools Reading", MA: Addison–Wesley, 1986.

[60] Buss, B., et al, "Investigating Reverse Engineering Technologies: The CAS Program Understanding Project". IBM Corporation, 1994.

[61] Harmon, P., "Objects In Action: Commercial Applications Of Object–Oriented Technologies", Jan, 1993. A–W ISBN 0–201–63336–1.

[62] Johnson, R., and Foote, B., "Designing Reusable classes", Journal of Object-Oriented Programming, Vol. 1, pp. 8–22, February 1989.

[63] Kim, W. and Frederick Lochovsky Editors, "Object-Oriented Concepts, Applications, and Databases", 1989.

[64] LaLonde, W. R. and Pugh, J.R., "Inside Smalltalk: Volume 1", Prentice Hall, 1990.

[65] Leffler, S.J., McKusick, M., Karels, M., and Quarterman, J., "The Design and Implementation of the 4.3BSD UNIX Operating System", Addison–Wesley, 1989.

[66] Madsen, O.L., Moller-Pedersen, B., and Nygaard K., "Object-oriented programming in the BETA programming language", Addison–Wesley, June 1993. ISBN 0 201 62430 3

[67] Mancoridis, S, and Holt, R.C, "Recovering the Structure of Software Systems Using Tube Graph Interconnection Clustering", Research PAper

[68] Martin, J. and Odell, J., "Object-Oriented Analysis and Design", Prentice–Hall, Englewood Cliffs, NJ., 1992.

[69] Ragland, B., "The Year 2000 Problem Solver, A Five-Step Disaster Prevention Plan", McGraw–Hill 1997.

[70] Rumbaugh, J., et al., "Object-Oriented Modeling and Design", Prentice Hall. 1991.

[71] Sciore, E., "Object Specialization", ACM Transactions on Information Systems, Vol. 7, No. 2, April 1989, p 103.

[72] Slinger, "Millennium bug terror stalks computer-land", an article, TORONTO STAR newspaper, page A2, Saturday October 11, 1997.

[73] Strachey, C., "Fundamental Concepts in programming languages", Lecture Notes for International Summer School in Computer Programming, Copenhagen, Aug. 1967.

[74] Stroustrup, E.M.A., "The Annotated C++ Reference Manual", Addison Wesley, 1990.