COMPUTER SUPPORTED ANALYSIS AND DOCUMENTATION TOOLS

AN EXAMINATION OF COMPUTER SUPPORTED ANALYSIS AND DOCUMENTATION TOOLS AND THEIR APPLICATION TO SYSTEMS DEVELOPMENT TECHNIQUES

By

JOHN WERNER GOOSSEN, B.A.

A Project
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Science

McMaster University

May 1981

MASTER OF SCIENCE
(Computation)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: An Examination of Computer Supported Analysis and Documentation Tools and Their Application to Systems Development Techniques

AUTHOR: John Werner Goossen, B.A. (Mathematics, University of British Columbia)

SUPERVISOR: Dr. R.J. Welke

NUMBER OF PAGES: vii, 176

ABSTRACT

The development of a "good" information system relies heavily on the designer's (or project leader's) understanding of the system's boundaries and environment within the organization, and the ability to derive an appropriate model of the system. This report presents an overview of the systems appproach to organizational theory and information systems development, and examines one computer supported analysis and documentation tool, PSL/PSA. The report also shows that this tool, PSL/PSA, is not only able to represent such a systems view, but that it is also valuable as an aid, in terms of modeling, to other techniques which try to reach these goals. The examination of PSL/PSA is followed with two examples showing how it is applied to the BIAIT BICMX and Structured Analysis.

ACKNOWLEDGEMENTS

I would like to acknowledge both R. Raskolnikov and J. Westerby for being true companions and providing diversion whenever necessary.

I would also like to express my appreciation to my fellow students in the Computation programme for the help and encouragement they provided throughout this time.

Finally, it is my pleasure to thank Dr. Richard Welke for the direction, encouragement, chicken-wings and beer he provided over the past year.

All of the aforementioned personalities have been instrumental in the consummation of this work and it is dedicated to them all.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

1.1 Information Systems Development

A major problem facing organizations is their attempt to develop a "good" information system (IS). The solution to the problem is dependent upon the definition of the word "good". Its meaning may vary from cost-efficient to customer-pleasing, labour-saving to employee-satisfying, or it may take on a larger scope in terms of benefits for the company and its community, or even mankind in general.

This problem of definition stems, at least in part, from the complexity of organizations as a whole. This complexity reveals itself in many ways, including the hierarchical division of personnel, the division of functions into departmental units (accounting, finance, marketing and production), and the tasks performed by the various functional units. The development of an information system, whether for the organization or one of its departmental units, must take into account the proposed systems boundaries and its surrounding environment. This development usually begins with the derivation of a model of the existing system. This model is then combined with requirements for the new system to form a model of the proposed system.

The development of an IS usually involves the search for a methodology or technique which allows the designer to develop an appropriate model of the system. All methodologies and techniques, whether they state it explicitly or not, derive their model from some predetermined view of the system. This view of the system may be characterized by a number of concepts such as whether a system is open or closed, what constitutes its boundaries and environment, how it interacts with its environment, what its goals are, and how these goals are achieved.

An analyst's view of the system development process and what it involves is, to a large extent, dependent upon how comprehensive a view is taken of the concept "system". The proposed system, and even the model of the existing system, will be limited in their accuracy, to the same extent that their conceptualization is limited in the mind of the analyst. A systems analyst (or any information systems developer) should be aware of the previously mentioned system concepts in order to arrive at a model which is large enough in scope to consider both the system and its surrounding environment. The attempt to model a system with consideration given to its environment, boundaries, and other factors, is known as a "systems approach".

The system developer has many aids available, all of which have specific ways of modeling the system. Inherent in most aids is also some sort of life cycle or ordered procedure which guides the analyst through a project from start to finish. The scope of this life cycle also has an effect on how the system is modeled.

Approaches to information systems development have evolved alongside the computer. The earliest approaches, such as flowcharts, were process and control oriented, whereas the variety of modern approaches range from improved flowcharting techniques to socio-technical systems designs [MUM79] which attempt to include the worker as part of the system being designed. Because systems development projects require much documentation, regardless of which method or technique is used, many aids include (or have as a by-product) the ability to document the system.

While most IS development approaches have progressed in terms of their broader systems outlook, some have also made use of the computer as an aid to the design and documentation process. Two such automated approaches are PSL/PSA (Problem Statement Language/Problem Statement Analyzer) [TEI76] and ERA/ERE (Entity Relationship Attribute/Entity Relationship Evaluator) [LIE80].

1.2 Overview

The focus of this paper will be an examination and application of one computer-aided systems documentation and analysis tool known as PSL/PSA. The examination of PSL/PSA will consider how it is used to describe a system, some of its report generating capabilities, and the advantages it has over other tools or techniques. After the initial examination, PSL/PSA will be applied to two analysis techniques - BIAIT [BUR80] and Structured Analysis [DEM78] - to determine if it may be used to enhance these techniques, as well as as a documentation tool in its own right.

The adequacy of PSL/PSA as a tool for the support of systems methodologies may be judged in two ways. These are its abilities to accomodate existing methodologies and its ability to maintain a broader conceptualization in anticipation of future developments.

PSL/PSA was chosen for a number of reasons. The major reasons include the fact that it is an automated technique, its claim to be amethodological, and its availability at McMaster. The majority of systems analysis aids are manual and many of those which are automated are simply computerized versions of the manual methods. PSL/PSA is different in that it evolved as part of a larger package meant for automatic program generation. One of the primary purposes of PSL/PSA is to describe information processing systems. This has also resulted in an effort to keep it from being tied to a particular methodology or technique.

The report begins with a summary of system theory. This is done to give the reader a broad base of reference for the following discussion on the application of system analysis techniques. System theory and its relationship to the development of a systems perspective of an organization is important because the view held by a systems developer will have a major effect on how the information processing system is designed.

With systems theory as a foundation, the next step will be to discuss a number of approaches and aids to systems development. Of these aids, PSL/PSA is singled out and examined in more detail. The ability of PSL/PSA to be used to model other techniques, in this case BIAIT and Structured Analysis, is then dealt with. Finally, some conclusions are drawn and recommendations are made about possible further investigations and applications.

CHAPTER 2

ORGANIZATIONAL SYSTEM THEORY

2.1 Systems Theory

From his earliest history to the present day man has attempted to make sense out of the seemingly hostile and chaotic world into which he has been born. The trend toward order and explanation has progressed from animism, through monotheism, to the present rationalistic and elementaristic view of scientific inquiry.

The physical sciences' (physics, astronomy, etc.) method of inquiry has been based on the idea of reducing all complex phenomena into elementary parts and processes as a way of explaining them. Ludwig von Bertalanaffy points out that: "this method works admirably well insofar as observed events were apt to be split into causal chains, that is, relations between two or a few variables...but questions of many variable problems always remained" ([BER72], p. 10). His dissatisfaction with this approach, as a way of explaining living organisms as a whole, led Bertalanaffy to the idea of a General Systems Theory (GST).

General Systems Theory began (circa 1925) with the attempt to view a living thing as a whole, teleological organism rather than a sum of individual parts and processes. The underlying premise of GST is derived from Aristotle's statement that "the whole is more than the sum of its parts" ([BER72], p. 9). Bertalanaffy, as the founder of GST, summarizes his view in this way:

> The properties and modes of action of higher levels are not explicable by the summation of the properties and modes of action of their components taken in isolation. If, however, we know the ensemble of the components and the relations existing between them, then the higher levels are derivable from the components. ([BER72], p. 10)

GST progressed to the point where those holding such a view expected it to replace the present approaches of empirical science by becoming the fundamental approach to science, whether social, biological or physical. To avoid association with the GST extreme, a number of terms such as systems thinking, system engineering, systemeering, etc. have been coined by those holding a more basic systems view. These views can be collected under the heading of the systems approach and are summarized by Mattessich as follows:

> This approach looks at systems holistically, emphasizing the interrelations of the systems' components as well as the properties and boundaries of the system vis-a-vis its environment. It ultimately focuses on the function or purpose of the system, and pays special attention to the hierarchy of systems as well as to the reconciliation of the goals of the super-system with those of the embedded subsystem. Finally it aims at the formulation of those features and laws that underlie the system in general. ([MAT78], p. 277)

The essence of systems theory is to view a system as a whole, in terms of its purpose, environment, parts and their interactions. This concept of a system applies to a wide range of objects throughout the universe. These types of systems can be divided into three distinct categories, which form a hierarchy. The lowest level consists of physical or mechanical systems, the next level is concerned with biological systems, and the third level is involved with human and social systems ([KAS70], p. 15). The concept of mechanistic systems includes static structures, simple dynamic systems with predetermined motions, and control mechanisms which are self-regulating to maintain system equilibrium. At the level of biological systems, life begins to be differentiated from not-life [KAS70]. Biological systems, which have a self-maintaining structure, are defined to include everything from cells to plants to animals, the latter of which are characterized by increased mobility, teleological behavior and self-awareness. The third level, human beings and social systems, are characterized by the previously mentioned points as well as the ability to utilize language and symbolism. This ability to use symbolism is expressed in such varied forms as messages, value systems, historical records, art, music, poetry, etc.[1]

Computer-based information systems may be classified in one of two ways. They are either seen as a simple "mechanical" system which accepts inputs and produces outputs, or they are seen from a broader perspective which includes not only the machine and its

---

1 This 3-level hierarchy is a consolidation of K. Boulding's nine level classification as outlined in [KAS70]

programs, but also the people using it, and the effects this use has on them and the organization. The view accepted by the analyst will determine how the system is developed.

2.2 System Description

Systems Theory allows a system to be described and analyzed as a whole rather than as a number of fragments. Even with the above summary of major systems' classifications, there is still a need to recognize some of the major ideas used in system description. The list of descriptors includes system boundaries, tendencies toward entropy and steady states, open and closed systems, equifinality and others. These descriptors will be discussed as they pertain to systems known as organizations.

2.2.1 Open and Closed Systems

With reference to systems, the concepts of open and closed refer to a system's relationship with its environment. Closed systems are those which are viewed without any consideration given to their environment. The lowest level in the hierarchy, i.e. physical and mechanical systems, may be considered closed systems. Biological and social systems, however, are thought of as open because of their interaction with their environment. Mattessich points out:

> . . that there is a theoretically important distinction between an open system, permitting inputs (matter, energy, information) from the environment and outputs (transformed

matter, energy, information) to the environment, and a closed system which is self-contained, permitting neither inputs from, nor outputs to, the environment. In practice the entire universe seems to be the only true closed system, hence the predominant interest, of systems research, in open systems. But occasionally, especially in the physical sciences, one may treat, for the sake of analytical convenience, an actually open system as though it were closed.([MAT78], p. 274)

This concept is one of the most important ones which arose from General Systems Theory, especially as it applies to organization theory. Traditionally, organizations have been seen as self-contained, closed systems whose internal structures, activities and relationships could be analyzed without reference to the surrounding environment. Approaching organizations, or any divisions therein, from an open systems viewpoint allows for the observation of inputs and outputs, with respect to their effects on the system. The result is a more intuitive and natural view of an organization. "For example, the business organization recieves inputs from the society in the form of people, materials, money, and information; it transforms these into outputs of products, services, and rewards to the organizational members sufficiently large to maintain their participation. For the business enterprise, money and the market provide a mechanism of the recycling of resources between the firm and its environment. The same kind of analysis can be made for all types of social organizations. Open-system views provide the basis for the development of a more comprehensive organization theory." ([KAS70], p. 19)

2.2.2 Entropy

The second descriptor, entropy, arises as a direct result of the discussion on open and closed systems. All closed systems have a tendency towards maximum entropy, as explained by the second law of thermodynamics. According to it, "a system moves toward equilibrium; it tends to run down, that is, its differentiated structures tend to move toward dissolution as the elements composing them become arranged in random order" ([KAT66], p. 91).

While this is true of closed systems, it is not true for open, i.e. biological and social, systems to the same extent. They do not move toward a random state without potential for work, but rather tend to maintain a steady state of dynamic equilibrium. To survive, open systems must move to arrest the entropic process; they must acquire negative entropy ([KAT66], p. 94). Open systems maintain themselves by accepting energy, in various forms, from their environment, transforming it internally and then expelling it in some altered energy form.

The fact that not all business organizations move toward a state of the highest random order is explained by seeing them as open systems within a particular environment. By taking more energy from the environment than they expend, systems may arrest the entropy process. In the case of biological systems this is not permanent, but social systems, with proper management of resources, can even progress toward order and organization. The slowing down and perhaps even reversal of the entropy process results in the open system achieving a steady state. This steady state is not the static equilibrium of closed systems but is a dynamic equilibrium, which is maintained by importing energy requirements to balance out the energy lost to the environment. There must be a constant recycling process of input, transformation, and output in order for the open system to survive.

2.2.3 Boundaries

The boundaries of open systems must be defined to distinguish them from their environment. It is important, in dealing with systems, to have a clear understanding of which objects and processes are part of the system, which interact with the system, and which are of no relevance whatsoever. Closed systems have boundaries which cannot be penetrated and therefore no consideration is given to their environment. This is acceptable for most physical and mechanical systems. As for biological and social systems, however, it is important to distinguish between the system and its environment because it is through interaction with its environment that the system is able to slow down the effects of entropy and maintain a dynamic equilibrium. The boundaries of biological systems, such as plants and animals, are easily identifiable in their environment. Organizations, however, because they are contrived and not natural, may have their boundaries arbitrarily set according to their functions and activities.

The scope and permeability of an information system's boundaries are dependent upon whether an open or closed systems view is adopted. With either view the system's boundaries may be set around the computer and its operators. The effect of setting such boundaries will depend upon the amount of consideration given to interactions between the system and it's environment through it's boundaries.

System boundaries for organizations need not be set at the physical level. An organization may be seen as having boundaries which set it apart within an economic, industrial or national environment. Similiarly, divisions within a company may be seen as systems within an organizational environment. Divisions can, in turn, be broken down into a number of departments, each with its own boundaries and environment. This classificaion, if continued to the bottom level, of groups consisting of employees, allows an organization to be viewed as a hierarchy of systems. Each system is therefore part of a larger supersystem and most systems are composed of subsystems.

2.2.4 Equifinality

Equifinality, a concept introduced by von Bertalanaffy

[KAT66], is another characteristic which distinguishes closed, physical systems from open, biological and social ones. The equifinality concept is in opposition to the direct cause and effect relationship found in closed systems, between the initial conditions and the final state. According to equifinality, a system can reach the same final state from differing initial conditions and by a variety of paths ([KAT66], p. 100). This concept is similiar to the Church-Rosser property (related to the correctness of parallel programs) as described by Rosen ([ROS73], pp. 160-161):

> In a Church-Rosser system, whenever one applies transformation rules to an object R (and then to the resulting object, and so on) until no further rules are applicable, the final result does not depend on which of several applicable rules was chosen at each stage.

The importance of this concept to the study of social systems is that it suggests that there is not one best way of achieving a desired outcome. Instead a satisfactory solution to a problem may be arrived at through a variety of paths and inputs. An extension of the concept of equifinality in systems is that of teleology.

Many open systems are teleological or goal-seeking in nature. This teleological nature is more than a simple tendency toward a steady state: it provides systems with characteristics such as purpose and choice. Russell Ackoff ([ACK71], p. 32) describes a goal-seeking system as follows:

> A **goal-seeking system** is one that can respond differently to one or more different external or internal events in one or more different external or internal states and that can respond differently to a particular event in an unchanging

> environment until it produces a particular state (outcome). Production of this state is its goal. Thus such a system has a **choice** of behaviour. A goal-seeking system's behaviour is responsive, but not reactive . . . Under constant conditions a goal-seeking system may be able to accomplish the same thing in different ways and it may be able to do so under different conditions.

The concept of goal-seeking is especially applicable to social organizations, as most of them do not exist simply for the sake of existence. They are usually formed with one of more specific goals in mind.

2.3 Organizations as Systems

Systems theory has grown from its original emphasis on biological organisms to encompass the majority of biological and social systems. As systems theory grew in prominence, traditional views of organizations as highly structured, closed system changed to more of an open systems approach ([KAS70], p. 18). The movement toward an open systems approach is borne out by a comparison of the modern organization to the previously listed attributes. The scenario of an organization as an open system which interacts with its environment, and attempts to achieve a high degree of order, has definable boundaries, has more than one way of getting things done (equifinality) and is goal-seeking in nature, is much more realistic than seeing an organization as a closed system with no environmental interaction.

Once such an open system view of organizations is accepted, there will also be a change in most functions of the organization, especially in leadership roles. The trend should be toward achieving an understanding not only of technical aspects of the organization but also of psychological, sociological and ecological aspects. If an open systems approach is advocated from the top levels of management on down, it should also have an effect on the way information systems are developed and implemented.

In this chapter the discussion has focussed on some ways of looking at information systems and the organizations which surround them. The process of developing or changing a system will be guided by the analyst's view of the system. Once an appropriate view of the system has been established (explicitly or implicitly), the system development process may begin. The next section focuses on a variety of information system development techniques and their relationships to the systems approach.

CHAPTER 3

SYSTEMS DEVELOPMENT PLANS

All attempts at system development are attempts to change the existing information system, however unrecognizable that system may be at the moment. Three perspectives from which system changes should be addressed are discussed in [WEL77]. They are systelogical, infological, and datalogical and are defined as ([WEL77], p. 150):

> SYSTELOGICAL perspective: How will changes to the existing information system alter/facilitate changes in the affected object systems [data processing system (DPS), user sub-system (USS)]?
>
> INFOLOGICAL perspective: How will changes to the existing information system alter/facilitate changes in the use of information by individuals who are members of the affected object systems?
>
> DATALOGICAL perspective: How will changes in the information system alter/facilitate changes in the data processing sequences associated with the DPS-USS object system intersection?

The development of any information processing system requires a plan which will take it through the stages of analysis, design and implementation. Development plans vary from organization to organization and range from ones which are explicitly laid out for the designer to those which are patched together as the applications arise. A number of approaches to information systems development have been proposed because of the apparent haphazardness of most development plans.

An open systems view may be applied to the development of an information processing system as well as to an organization. Such a view is necessary if the newly designed system is to fit into its environment with the least amount of friction. Systems approaches to the development of information systems are known as systems development plans. The majority of these development plans are cyclic or sequential in nature and fall into the category of systems life cycles. Some of these system life cycle approaches will be discussed in this chapter, as will the contrasting Finnish PSC Systemeering Theory [KER79b].

3.1 System Life Cycle

System life cycle (SLC) is a generic term applied to a wide variety of problem solving methods which begin with some sort of preliminary study and end with the implementation of the proposed solution. The number of steps for most of these methods falls into the 7 $\pm$ 2 range. The "life cycle" refered to in these methods is that of the information system being designed, or the problem being solved. Figures 3.1-3.7 (taken from [ALT80], [KOB72], [CHE72], [LUC76], [MUM79], [BAS78] and [ALT80] respectively) show examples of some problem solving methods. None of these are representative but all give an idea of the stages passed through en route to designing an information processing system.

The life cycle approach can be viewed in a variety of ways. Four possible ways of seeing it are outlined by Koberg ([KOB72], p. 20-21) in The Universal Traveler:

> **Linear:** Where one thing follows another in a straight line.
>
> **Circular:** Where there is continuity, but never a beginning or end. As one problem situation appears to be resolved, another one appears to begin.
>
> **Feedback:** Seeing it as a constant feedback system where you never go forward without always going back to check on yourself; where one progresses by constant backward relationships and where the stages of the process advance somewhat concurrently until some strong determining variable terminates the process (time, money, energy, etc.)
>
> **Branching:** Others see the design process as a branching system where certain events determine more than one direction and where directional progress is achieved via a many-branched excursion.

As seen in Figs. 3.1-3.7, there are a variety of approaches to information system problem solving, but all seem to go through the same basic phases. The Lucas (Fig. 3.4), Checkland (Fig. 3.3), and Kolb-Frohman (Fig. 3.7) methods define their approaches specifically in terms of developing information processing systems. Checkland [CHE72] does take into account a wider system and its environment, calling it a 'purposeful human activity system'. As seen in Fig. 3.3, he also makes use of a feedback mechanism. Koberg (Fig. 3.2), on the other hand, describes a general problem solving method which attempts to involve feelings and creativity. Mumford (Fig. 3.5) tries to incorporate human or sociological aspects into the design process. The Lewin-Schein Theory (Fig. 3.1) describes three general stages of change. Both Barrow's Clinical Method (Fig. 3.6) and the Kolb-Frohman Model realize the need for interaction between the problem solver and the one with the problem.

The similiarities can be seen in that all, at some point, go through the stages of observation and definition of the problem, checking requirements and changes needed, designing alternatives and selecting the best, and implementation and review of the selected solutions. The majority of SLC approaches, as applied to information systems development, begin with a datalogical perspective. Exceptions to this are Mumford [MUM79] and Checkland [CHE72] who attempt to reach a systelogical perspective.

The effectiveness of any of the SLC approaches is that they break down the rather large problem of systems development into a number of smaller tasks or subparts. In this way a problem may be solved one step at a time. This breakdown of the problem solving process, while it does make the problem easier to solve, may do so at the expense of losing the "systems view" discussed in the first chapter. If this is the case, the problem being solved will not be the original problem.

LEWIN-SCHEIN THEORY OF CHANGE

UNFREEZING: Creating an awareness of the need for change and a climate of receptivity for change.

MOVING: Changing the magnitude or direction of the forces that define the initial situation; developing new methods and/or learning new attitudes and behaviors.

REFREEZING: Reinforcing the changes that have occured, thereby maintaining and stabilizing a new equilibrium situation.

FIGURE 3.1

KOBERG'S LOGICAL SEQUENCE OF DESIGN PROCESS

ACCEPT SITUATION: To find reasons for going on: to state initial intentions. To accept the problem as a challenge; to give up our autonomy to the problem and allow the problem to become our process.

ANALYSE: To get the facts and feelings: to get to know about the ins and outs of the problem; to discover what the world of the problem looks like.

DEFINE: To determine the essential goal(s): to decide what we believe to be the main issues of the problem; to conceptualize and to clarify our major goals concerning the problem situation.

IDEATE: To generate options for achieving the essential goal(s): to search out all the possible ways of getting to the major goals. Alternatives.

SELECT: To choose from the options: to compare our goals as defined with our possible ways of getting there. Determine best ways to go.

IMPLEMENT: To take action (or plan to act): to give action or physical form to our selected "best way".

EVALUATE: To review and plan again: to determine the effect of ramifications as well as the degree of progress of our design activity.

FIGURE 3.2

CHECKLAND'S SYSTEM BASED METHODOLGY

```
---------> 1. ANALYSIS
| | | |       1.1 Examine the problem situation and collect
| | | |           candidates for the role "the problem".
| | | |       1.2 Analyse the problem situation. (Structure,
| | | |           Process, Relationship between them.)
| | | |
---------> 2. ROOT DEFINITION OF RELEVANT SYSTEMS
| | | |       2.1 Formulate root definitions of relevant systems.
| | | |
| | | |    3. CONCEPTUALIZATION
| | | ^       3.1 Assemble the minimum necessary activities in the
| | | |           system(s) 2.1, hence build conceptual models.
| | ^ |       3.2 Use the 'formal system' concept and/or other
| | | |           systems thinking to finalize the
| | | |           conceptualization(s).
| ^ | |
| | | |--- 4. COMPARISON AND DEFINITION
| | |         4.1 Make a formal comparison between the results of
| | |             1. and 3.
| | |         4.2 From the results of 4.1 define a range of
| | |             possible changes.
^ | |
| | |----- 5. SELECTION
| |           5.1 Select, with relevant actors in the problem
| |               situation, or get them to select, a relevant
| |               feasible change required to improve the
| |               situation in 1.2.
| |
| |------- 6. DESIGN AND IMPLEMENTATION
|             6.1 Design whatever is necessary for the
|                 implementation of the change selected.
|
|--------- 7. APPRAISAL
```

FIGURE 3.3

LUCAS'S SYSTEMS ANALYSIS AND DESIGN STAGES

MOTIVATION

The existing system
Preliminary survey of system objectives, requirements
Decision to proceed with feasibility study

FEASIBILITY STUDY

Sketch existing procedures or system
Formulate rough alternative system
Estimate costs
Decision to proceed with system

SYSTEMS ANALYSIS

Detailed study of present appraoch
Collection of data on volumes, input/output, files

AN IDEAL SYSTEM

System unconstrained by cost
Revisions to the ideal system to make it acceptable
Decision on systems alternatives

DETAILED SPECIFICATIONS

Processing logic
File design
Input/output
Programming requirements
Manual Procedures

IMPLEMENTATION

Programming and testing
Documentation
Training
Conversion
Installation

FIGURE 3.4

MUMFORD'S SOCIO-TECHNICAL SYSTEM DESIGN

Social System Analysis

Set Social Objectives

Specify Social Alternatives

Technical System Analysis

Set Technical Objectives

Specify Technical Alternatives

Socio-Technical Analysis

Match as socio-technical alternatives

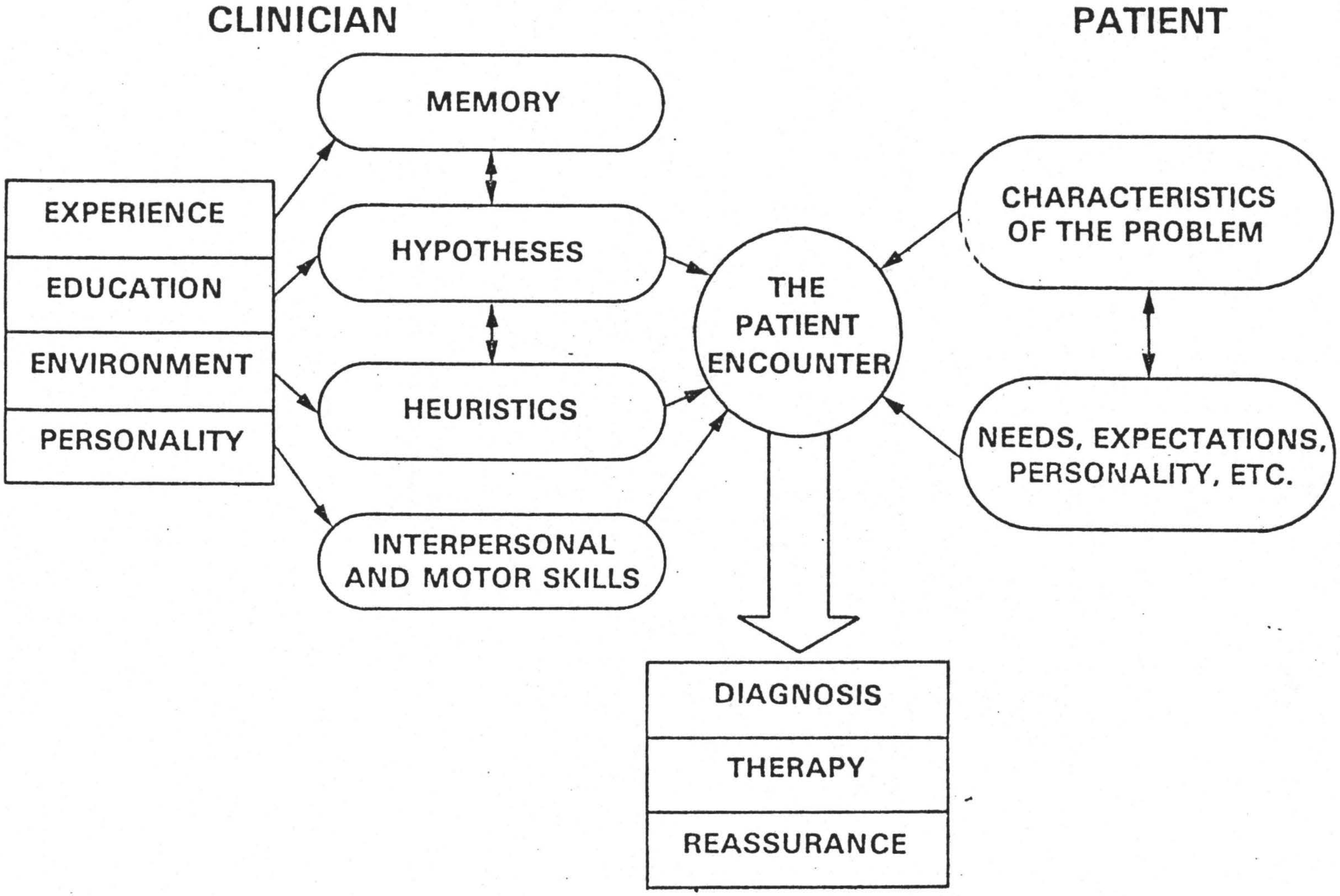
Rank in terms of ability of each alternative
to meet social and technical objectives

Consider Costs/Resources/Constraints

Select best socio-technical solution

FIGURE 3.5

A MODEL OF THE CLINICAL METHOD



BARROW'S MODEL OF THE CLINICAL METHOD
FIGURE 3.6

KOLB-FROHMAN CONSULTING PROCESS MODEL

SCOUTING: User and designer assess each other's needs and abilities to see if there is a match; an appropriate organizational starting point for the project is selected.

ENTRY: User and designer develop an initial statement of project goals and objectives; commitment to the project is developed; user and designer develop a trusting relationship and a "contract" for conducting the project.

DIAGNOSIS: User and designer gather data to refine and sharpen the definition of the problem and goals for the solution; user and designer assess available resources (including commitment) to determine whether continued effort is feasible.

PLANNING: User and designer define specific operational objectives and examine alternative ways to meet these objectives; impacts of proposed solutions on all parts of the organization are examined; user and designer develop an action plan that takes account of solution impacts on the organization.

ACTION: User and designer put the "best" alternative into practice; training neccessary for effective use of the system is undertaken in all affected parts of the organization.

EVALUATION: User and designer assess how well the goals and objectives (specified during the Diagnosis and Planning stages) were met; user and designer decide whether to work further on the system (evolve) or to cease active work (terminate).

TERMINATION: User and designer ensure that "ownership" of and effective control over the new system rest in the hands of those who must use and maintain it; user and designer ensure that necessary new patterns of behavior ahve become a stable part of the user's routine.

FIGURE 3.7

Although there are some problems with the SLC approach, if a complete systems view can be maintained (but not necessarily by the analyst) while progressing through each stage of the cycle, it can make the problem solving more complete. The SLC approach is the most common one in use at this time, but there are other attempts at problem solving which take a different approach. The most notable of these is the Finnish PSC-Model. An attempt is also being made at developing a meta-framework which permits comparison of different design methodologies against common criteria (see [KLE80]).

3.2 Finnish PSC-Model

The Finnish PSC Systemeering Theory [KER79b] approaches the development of information systems from a general systems viewpoint. The stress is on an infological approach which emphasizes the investigation of people and organizations utilizing information, influences of the data system on its environment, and other factors ([KER79b], p. 2). Rather than partition the life of an information processing system into a number of stages, the development process is seen as a hierarchy of basic structures or contructs which are used to build up the different levels of the PSC model ([KER79b], p. 6).

The process of systems development is seen as having three main aspects - pragmatic (P), semantic (S), and constructive (C).

These aspects may be characterized as follows:

> The Pragmatic Aspect: considers the environment surrounding the target system, its needs in relation to the target system and the effects on it as a result of any output from the target system.
>
> The Semantic Aspect: considers the target system as a black box. It studies only the external behaviour, in terms of the system's required inputs and resultant outputs.
>
> The Constructive Aspect: is a combination of two aspects. The internal behaviour of the target system is studied and the target system is also studied in light of its functions and the resources necessary for these functions.

Once the three main aspects have been defined the PSC-Model is built up according to the following constructs ([KER79b], p.6):

- structure of development and control function
- structure of learning and knowledge basis
- subsystem structure of data system
- main functions of purposeful system
- discontinuity of process

The hierarchical PSC structure is the result of combining the three main aspects and the partition of the development function into design and test stages (see Fig. 3.8). Kerola explains the hierarchical structure of the figure in the following way ([KER79b], p. 8-9):

> The vertical lines in Fig. 2.5 refer to the sequence of performance in a successful case, and the horizontal lines to feedback loops. It can be represented in sequence form in the following way:
>
> (I) $P_D\ S_D\ C_D\ B\ C_T\ S_T\ P_T$
>
> According to the sequence I, we have the following tasks:

P_D = pragmatic design
S_D = semantic design
C_D = constructive design
B = implementation of elements
C_T = constructive test
S_T = semantic test
P_T = pragmatic test

in this order. The testing tasks will be performed in the opposite order from the design task, for the natural reason that it is not sensible to test the semantic feature of the result unless the construction of the result is feasible, nor profitable to test if the result has the desired benefits (P_T), unless it is semantically feasible.

```
P  <-------------------------------------------  P
 D  ------------------------------------------->  T

   |                                           /|\
  \|/                                           |

      S  <-----------------------------------  S
       D  ----------------------------------->  T

         |                                 /|\
        \|/                                 |

            C  <-------------------------  C
             D  ------------------------->  T

               |                       /|\
              \|/                       |

                          B
```

THE HIERARCHICAL PSC STRUCTURE
FIGURE 3.8

Even this brief overview of the PSC model and method allows one to see that it is quite different from the SLC approach to systems development. A comparison of these two approaches as well as an evaluation of the PSC systemeering theory can be found in [KLE80].

CHAPTER 4

AIDS TO SYSTEMS DEVELOPMENT

Regardless of which of the previously mentioned approaches to systems development are followed, there are aids available to carry out any plans. The plan or procedure followed in developing an information processing system must have some degree of structure because of the specific nature of the end product. The appropriate degree of structure is dependent upon the degree of certainty required with respect to the results desired, i.e. the development of a Decision Support System (DSS) [KEE78] would require a less structured plan than the development of a bookkeeping system. Besides being structured, most plans also require a lot of documentation at every stage of development.

There are a number of aids available, ranging from handbooks which take the design team through the complete life cycle (e.g. IBM's Project Management Guide) to proprietary techniques which may be applied to only one aspect of the overall development plan [e.g. Don Burnstine's BIAIT (Business Information Analysis and Integration Technique)]. There are also documentation tools available, both manual (HIPO, SADT) and automated (PSL/PSA, ERA/ERE).

Aids to systems development can be divided into three categories: methodologies, techniques, and tools. These categories are distinguished by the following definitions:

> Methodology: is a general way of doing things. A method or process may be described but the steps are vaguely defined, so they may be carried out in a variety of ways.
>
> Technique: is a specific way of performing a task. A prescribed procedure is followed with each step well outlined.
>
> Tool: is something that serves as a means of fulfilling a task. A tool may be used in a number of ways, independent of any technique.

The majority of aids available are aimed at helping the analyst during the analysis and design stages of the development plan. The evolution of systems analysis aids has lagged approximately one generation behind the evolution of computing devices. An increased awareness of the need for aids has reduced this gap considerably [COU73]. This awareness, along with the progress in computer technology, has resulted in some automated aids being developed in order to exploit the computer to its full advantage.

4.1 Approaches

Any attempt at helping an analyst in the development of an information system requires a way of modelling the system. The building of this model and the manipulation thereof is important because it allows the analyst to conceptualize the system as a whole. One should also be aware that a model is not a representation of the system, but is based on the modeler's perception of the system. As a result, one system may be modeled or seen in many different ways.

The approaches, in terms of a systems view, taken by most aids can be divided into two groups: data-centred and process-centred. In the data-centred approach, the information system is made up of many pieces of data and the changes which occur to the data are the relationships which bind it into a system. In the process-centred approach, the information system is made up of a number of processes and the data passed between them forms the relations which tie the system together. The majority of approaches currently in use are process-centred.

One of the earliest techniques (1950's) for analyzing and documenting computer based systems was the flowcharting method. The system was layed out graphically through the help of a number of outlines and symbols. While these charts did enhance the communication between analysts and programmers, their hardware and process orientation, and lack of standardization, among other things, paved the way for a new generation of techniques (see [COU73] for a history of systems analysis techniques).

The step beyond unorganized flowcharting has been structured charting techniques such as ADS, SOP, HIPO, etc. The general rhythm of these approaches is to document the existing, physical data processing system and then develop an equivalent document for a proposed alternative, where the alternative reflects technological substitutions for processes, re-organization of data and the inclusion/revision for requested reports ([WEL77], p. 154). More recent approaches (1975-80) have attempted to go beyond the datalogical perspective to achieve an infological or even systelogical perspective. Such approaches include SADT, structured system analysis methodologies ([DEM78], [GAN79], [MEN80]) and others. Their attempt is to cover the major portion of the life cycle from analysis of the existing system to the beginning of program design. There also exist techniques whose aim is to analyze one specific phase of the life cycle. Most notably are front end techniques such as IRA (requirements elicitation) and BIAIT or BSP (planning).

Through the increased capabilities of computers and the growth of large information systems, it became apparent that the computer could be used as an aid in the systems development process. The earliest attempts were to automate existing manual techniques such as IBM's TAG and NCR's ADS. Although beneficial, the automating of manual system analyzing techniques produced suboptimal results ([COU74], p. 205).

While manual systems were being automated, theoretical work was also being done by Young and Kent, Grindley, Langefors, and CODASYL [COU74]. Some of the more recent automated techniques include ISAP, SAMMDF, SREM and others. A survey of some of these automated aids has been compiled [TRE80].

There are many other development aids in existence; some are available commercially, others through research projects, and still others developed by companies for their internal use. Most approaches have their merits but they are confining in their systems view. Recently, attempts have been made at developing computer-aided tools which are able to provide a wider systems view. In this paper these are given the name "generalized-aids".

4.2 Generalized-aids

A generalized-aid may be defined as a tool which, rather than tying the analysis and design process to a particular methodology or technique, allows the system to be described according to some given criteria and then allows the system view to be altered as the system is altered. The generalized-aid should go beyond a datalogical perspective and approach the system description from an infological or even a systelogical perspective. If the general aid has too narrow a perspective, it risks becoming just another technique.

In order to achieve this perspective, a system model or description must be arrived at using general system terms or categories. All systems may be described using three categories - objects, properties and relationships. A system is made up of objects, each object has a number of properties, and the objects are interrelated through one or more relationships. This concept is approximately equivalent to the Entity/Relationship/Attribute concept (see [CHE80]) depending upon whether relationships can have properties or not.

The rest of the chapter will concern itself with two generalized-aids, PSL/PSA and ERA/ERE. These two were chosen because they are both automated and are able to describe the system using general system terms.

4.2.1 PSL/PSA

PSL/PSA is the result of research conducted under the direction of Prof. D. Teichrow, as part of the ISDOS project, at The University of Michigan. It is a software package meant to be used as an aid for systems analysis and documentation of both existing and proposed systems. Although it is most commonly used in conjunction with information systems development, it can be used to describe and document any type of system (e.g. production, marketing, biological).

At the highest level, PSL/PSA takes the form of any modern data processing system. At the center of the system is a data base into which the system description is entered and maintained. The use of PSL/PSA is divided into two stages. The system is described using a non-procedural language called Problem Statement Language (PSL). Each object is defined along with its properties and the relationships in which it takes part. An example (taken from the Structured Analysis application) of a number of PSL object definitions can be seen in Figure 4.1.

Once the system has been described, this desription is submitted to the Problem Statement Analyzer (PSA). PSA checks for a consistent description, and updates the analyzer database accordingly. To extract information from the stored system description in the data base one can use either an ad-hoc type of query language or use one of a number of report generator modules, each presenting certain aspects of the system in a predefined way, ranging from indented lists to network diagrams. An overall view of PSL/PSA can be seen in Figure 4.2 ([TEI76], p. 44).

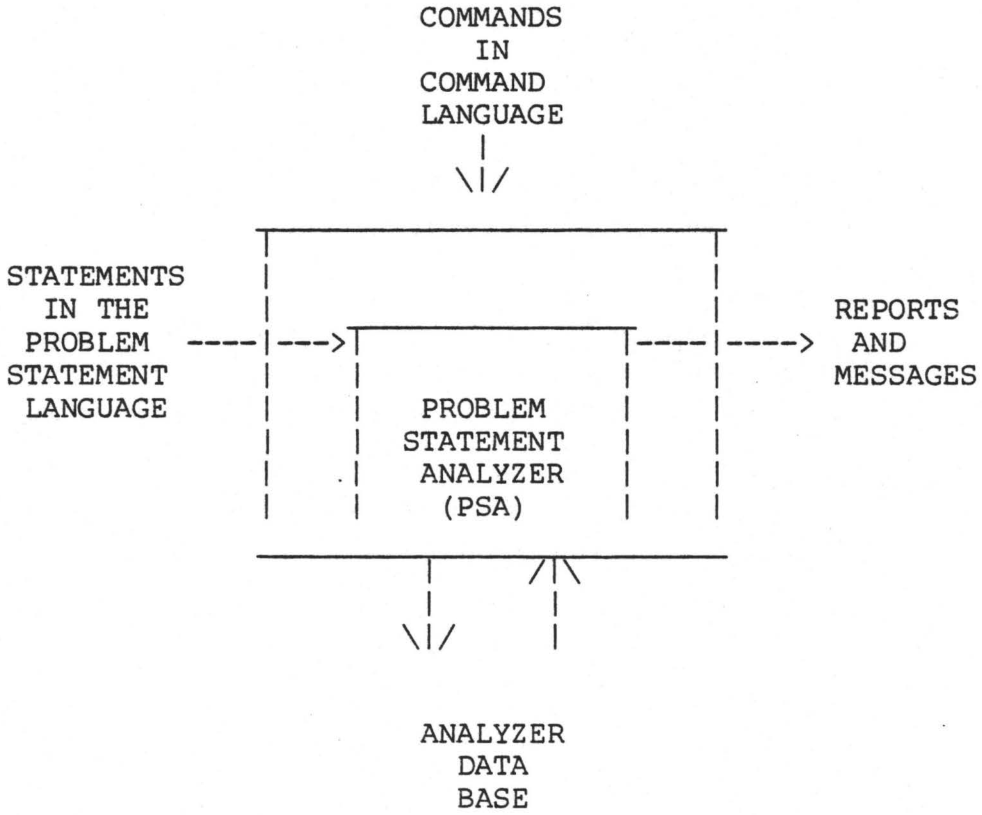
4.2.2 ERA/ERE

ERA/ERE [LIE77] is also a computer aided tool meant to be used for the documentation and analysis of information systems. It has been developed by Arthur Z. Lieberman, and is used to describe a system in terms of Entities, Relationships and Attributes, hence ERA. The second part, a tool called ERE (Entity/Relationship Evaluator) allows a designer or documentor of an information system to query the contents of the design database and to evaluate the quality of the documented design using a

```
PSL TERMS             SPECIFIC OBJECT SYSTEM INSTANCES

DEFINE PROCESS              Arrange-delivery;
       RECEIVES                     Send-list;
       PART OF                      Dispatch-goods;
       DERIVES                      Delivery-ticket
         USING                        Send-list, Sold-order;

DEFINE ENTITY               Sold-order;
       DESCRIPTION;
         This is the copy of the sale document used for dispatch.;
       CONSISTS OF                  Clean-order-copy2,
                                    White-ticket-copy;
       DERIVED BY                   Record-order
         USING                        Clean-order-copy2,
                                      White-ticket-copy;
       USED BY                      Arrange-delivery
         TO DERIVE                    Delivery-tickets;

DEFINE GROUP                White-ticket-copy;
       CONSISTS OF                  Hanger-ticket-form,
                                    Buyer-ID,
                                    Negotiated-amount;
       CONTAINED IN                 Filled-in-tickets,
                                    Sold-order,
                                    Retail-receipt;
       DERIVED BY                   Record-order
         USING                        Remaining-ticket-copies;
       USED BY                      Record-order
         TO DERIVE                    Sold-order;

DEFINE ELEMENT              Buyer-ID;
       DESCRIPTION;
         This is a seven digit buyer identification code.;
       CONTAINED IN                 Receipt,
                                    Clean-order-copy1,
                                    Clean-order-copy2,
                                    Red-ticket-copy,
                                    White-ticket-copy;

DEFINE INPUT                Send-list;
       DESCRIPTION;
         This is the list of dispatching instructions.;
       GENERATED BY                 Caisse-Central;
       RECEIVED BY                  Wholesale-operations,
                                    Dispatch-goods,
                                    Arrange-delivery;
       CONSISTS OF                  Wholesaler-ID,
                      One-or-more   Orders-to-dispatch;
       USED BY                      Arrange-delivery
         TO DERIVE                    Delivery-tickets;
```

EXAMPLE PSL OBJECT DEFINITIONS

FIGURE 4.1



THE PROBLEM STATEMENT ANALYZER

FIGURE 4.2

"rules" language ([LIE80], p. 641).

4.2.3 Comparing ERA/ERE with PSL/PSA

There are many similiarities between PSL/PSA and ERA/ERE, as well as some marked differences. Both aim to document and analyze a system through the use of a stored description and information retrieval. PSL/PSA has been in various stages of development since 1971 and has been in a relatively stable state since 1976. Because it is a university research project, its evolution continues at The University of Michigan as well as several other universities and some private institutions. PSL/PSA has been adapted by a number of organizations, e.g. A.T.& T. Long Lines, Boeing, Chase Manhattan Bank, British Railways, Rockwell International, IBM, Jet Propulsion Laboratories, Texas Instruments, The Royal Bank of Canada, and the Canadian Department of National Defence. ERA/ERE was first implemented in 1977 and re-implemented in 1979. It is currently in use at Bell Laboratories.

ERA and PSL both allow systems to be described in terms of the object/property/relationship criteria described above. PSL is much more limiting, however, in its descriptive power, because of its lack of generality, with respect to naming object, property, and relationship types. PSL has limited the number of predefined object types to 18 and has also specified what types of properties and relationships each object type may have. ERA, on the other hand, allows any number of entities and relationships, and the attributes to be defined. Both ERA and PSL provide syntax and consistency checking for their entries.

The reporting and analysis aspects of both ERE and PSA attempt to reach similiar goal through various paths. ERE extracts information from the data base via a query language called RULES. RULES allows the extraction of "entity-relationship (ER) strings" which describe chains of entities that are related to one another ([LIE80], p. 646). Queries may also be made which list relationships associated with a particular entity-type. A second feature of RULES is that it allows the description to be checked for any desired completeness or consistency. Completeness checks may be made to see whether all processes are defined, or if there are a given number of attributes defined for every process. Completeness checks could include queries to determine whether a process sends parameters to itself, or if a chain of process priorities are in the proper order.

PSA, on the other hand, has a number of predefined reports available which present information in lists, tables and matrices. These reports are used to summarize information, check for consistency, present graphical views of the system and report on data usage. Aside for the predefined reports, PSA also allows for the extraction of data from the stored system description through the use of an ad-hoc type of query language.

In this section we have examined two generalized-aids, PSL/PSA and ERA/ERE, which have both utilized the computer to advantage in terms of description storage, consistency checking and report generating. These comparisons are meant to give an idea of the similiarities and differences between PSL/PSA and ERA/ERE and not as an evaluation. PSL/PSA will now be examined in depth and then its application to two different system development aids will be discussed.

CHAPTER 5

PSL/PSA

5.1 Overview of PSL/PSA

PSL/PSA may be used to document a system or any model of a system. To use it in a specific environment the first and most important step is to learn how to "see" and describe a system in terms of the objects, properties, and relations which PSL uses. This is mainly a problem of semantics. For example, the common term for a processing routine in one organization or technique may be the term "module" or "task" or "program" or "activity". PSL uses the term PROCESS which can in turn be hierarchically broken down into sub-processes which are called sub-parts of the process. Similarly, the terms files, records and items have their equivalences in PSL as SETS, ENTITIES and ELEMENTS.

Determining and naming the object types present in the system is only half the problem. The other half is that of describing the properties and relationships associated with each object type. Property descriptors are used to uniquely identify an object of a group of objects. Such descriptors include KEYWORD, TRACE-KEY and user defined ATTRIBUTES. Relationships between objects are defined though the use of verbs such as

MAINTAINS, DERIVES, EMPLOYS, USES, UPDATES, etc. These verbs may be used to describe relationships between two, three, and even four object types. It is therefore important to know exactly how each verb is used and what results this usage will have on the reports generated.

For each organization, methodology, and technique, there is a systems nomenclature for describing system objects and relations which must be translated into PSL terms. Once this is done, however, the representation of systems in PSL/PSA is straightforward. In this chapter we take a closer look at PSL/PSA, to see how it is used to document a system and what sort of reports can be obtained.

5.2 Problem Statement Language

PSL is a formal language used for describing systems; it is not a procedural or programming language. In order to describe a system using PSL, it is important to understand the model of a general system upon which it is based. The model breaks the system into three basic components. First, the system consists of things which are called OBJECTS. Second, each object has PROPERTIES by which it may be defined. Third, objects may be connected or interrelated in various ways called RELATIONSHIPS. In PSL, this general model is specialized for an information system by allowing the use of only a limited number of predefined object, property, and relationship types ([TEI76], p. 44).

PSL allows for 18 different object types to be used in defining a system (FIGURE 5.1). There are three major categories of classification for the object types. They are communication and analysis aids, organizational units, and target system descriptions (FIGURE 5.2).

Object types ATTRIBUTE, MEMO and SYSTEM-PARAMETER are classified as **communication and analysis aids.** Although more accurately seen as properties themselves, these object types are used to highlight special properties of certain objects in the system. They may be used in cases where the analyst wishes to include some information in the documentation which cannot be easily expressed in the formal syntax. ATTRIBUTES and their values can be used to describe particular objects in the system that may not be described by any other PSL statement. A MEMO is a text (comment) entry which may apply to more than one object in the system. A SYSTEM-PARAMETER is used to represent a value (or range of values) which is relevant to characterizing system size.

Organizational units or objects are those which in some way interact with the target system but are not a direct part of it. This object type is known as an INTERFACE. It helps to define the boundaries of the system. Regardless of the complexity of its internal functions, only the description pertaining to its

SUMMARY OF OBJECT TYPES

Object Type	Synonyms
ATTRIBUTE	ATTR, ATTRIBUTES
CONDITION	COND, CONDITIONS
ELEMENT	ELE, ELEMENTS
ENTITY	ENT, ENTITIES
EVENT	EV, EVENTS, EVT
GROUP	GR, GROUPS
INPUT	INP, INPUTS
INTERFACE	INTF, ORGANIZATIONAL-UNITS, ORGU, ORGANIZATIONAL-UNIT, REAL-WORLD-ENTITIES, RWE, REAL-WORLD-ENTITY, INTERFACES
MEMO	MEMOS
OUTPUT	OUT, OUTPUTS
PROCESS	PROC, PROCESSES, PRC
PROCESSOR	PROCR, PROCESSORS, PRCR
RELATION	RLN, RELATIONS
RESOURCE	RSC, RESOURCES
SET	SETS
SYSTEM-PARAMETER	SYSP, SYSTEM-PARAMETERS, SYSPAR
UNIT	UNITS
UNDEFINED	

FIGURE 5.1

relationship with the target system are important. An INTERFACE is defined where the environment receives information from, or generates information for, the target system.

The third and largest group of object types are those used to **describe the target system.** These types are divided into seven categories (FIGURE 5.2):

1. Collection of information.
2. Collection of information instances.
3. Relationships among collections of information.
4. Data definition.
5. Data derivation.
6. Size, volume and resources.
7. Dynamic behavior.

A **collection of information** is thought of as information which is related to one particular type of thing or concept. There are three types of collections of information that can be defined in PSL; INPUTS, OUTPUTS, and ENTITIES. The differences among these types of collections are related to their role in the target system. An INPUT is a collection which is produced outside of the system, but it is used by the system in some way. An OUTPUT is a collection which is produced by the system but it is used outside of the system. An ENTITY is maintained internal to the system. ENTITIES are initially derived and maintained using information from INPUTS. OUTPUTS are then derived using information from ENTITIES.

The object type SET is used to define a number of instances of one or more collections of information. A SET can be thought

CLASS OF OBJECT TYPES	OBJECT TYPES
COMMUNICATION AND ANALYSIS AIDS	ATTRIBUTE MEMO SYSTEM-PARAMETER
INTERFACES OR ORGANIZATIONAL UNITS	INTERFACE
TARGET SYSTEM	
COLLECTIONS OF INFORMATION (EXTERNAL) (INTERNAL)	 INPUT OUTPUT ENTITY
COLLECTIONS OF INFORMATION INSTANCES	SET
RELATIONSHIPS AMONG COLLECTIONS OF INFORMATION	RELATION
DATA DEFINITION	GROUP ELEMENT
DATA DERIVATION	PROCESS
SIZE, VOLUME AND RESOURCES	PROCESSOR RESOURCE UNIT
DYNAMIC BEHAVIOR	EVENT CONDITION
OTHER	UNDEFINED

CLASSIFICATION OF OBJECT TYPES

FIGURE 5.2

of as a file which contains records. These records are the object type ENTITY. A SET may also consist of INPUTS or OUTPUTS.

Collections of information maintained internal to the system (ENTITY) are often related to each other in that there is information which is not inherent to either yet is associated with both. This kind of logical connection among ENTITIES can be described using the object type RELATION.

Collections of information (INPUTS, OUTPUTS and ENTITIES) contain values of information called ELEMENTS and GROUPS. ELEMENTS are the basic unit of information and cannot be subdivided. They are used to describe a data object which may take on a value, i.e. name, age, etc. GROUPS are used to describe a collection of ELEMENTS and/or GROUPS. The use of GROUPS is primarily a notational convenience which allows the problem definer to logically relate one or more ELEMENTS and/or GROUPS together under one GROUP name.

The purpose of an information processing system is to process data, i.e. to produce data values from other data values. In PSL, the object type used to describe this transformation is PROCESS. A PROCESS is defined by specifying the information upon which it operates and the information which it produces.

UNIT, RESOURCE and PROCESSOR are object types which help describe the system in terms of the size of the system, volume of work the system will do and resources which it will use. UNITS are used to describe the frequency of an occurence in the system, such as "year" or "work week". A RESOURCE is an object that the physical elements in the target system (PROCESSORS) consume in order to carry out information processing functions. RESOURCES may be measured in UNITS. Once an amount of a RESOURCE is consumed it is considered unrecoverable because it is "used up". An object that can "perform" a PROCESS is a PROCESSOR. A computer, a department or a person can all be defined as PROCESSORS.

Dynamic behavior of the system can be described through the use of object types EVENT and CONDITION. An EVENT is used to describe possible occurences during the operation of the target system. An occurence of an EVENT is associated with a specific point in time although it may occur more than once during system operation. A CONDITION is used to describe some state of the target system.

The above passage shows that PSL allows a variety of object types to be defined in the description of a target system. Once an object type is defined, it can be described in terms of its properties and relationships with other objects. The properties and relationships allowed in PSL are grouped into eight major categories on the basis of the aspect of the system which they describe. Aspects which express property information and relationship information are denoted by a (P) and (R) respectively. The eight major aspects are:

1. **Communication and Analysis Aids** (P). The properties defined in this section may be used to distinguish objects of the same type.

2. **System Boundary and Input/Output Flow** (R). The System Flow aspects of the system deal with objects (INPUTS, OUTPUTS) which come from and are supplied to the environment (INTERFACES).

3. **System Structure** (P). System Structure is concerned with the hierarchies inherent in most types of systems.

4. **Data Structure** (P). Data Structure represents the relationships that exist among data used and/or manipulated by the system, as seen by the users of the system. It is also used to describe the way data is grouped in collections of information.

5. **Data Derivation** (R). This section specifies what information is used, updated and/or derived, how this is done and by what processes. Whereas System Flow deals only with high level collections of information, which are input to and output from the system, Data Derivation can deal with the very lowest transformation of data.

6. **System Size, Volume and Resources** (P). The System Size is concerned with those factors which influence the volume of processing that will be required. To describe system size, the parameters involved are named as objects.

7. **System Dynamics and Control** (P). The dynamic analysis aspect of the system description presents the manner in which the system behaves over time.

8. **Project Management.** Project Management properties are used to provide information about the individual or group writing the PSL/PSA description.

5.2 Problem Statement Analyzer

PSA is a software package which accepts the system description written in PSL, modifies the data base accordingly and on demand produces reports which help to analyze and "see" various aspects of the system.

PSA consists of four main parts. The first is a Command Language Interface which analyzes statements and executes other routines as required. The second is an analyzer which checks statements written in PSL. The third is a report generating facility which produces a given report on instruction from the Command Language Interface. The fourth is a data base management system which provides the interface between PSA and the data base [ISD79].

The main feature of PSA of interest to the Information System developer is its report generating facilities. There are about 20 reports which may be generated (FIGURE 5.3) and they can be classified according to the purposes they serve ([TEI76], p. 44):

> Stored System Description (Data Base) Modification Reports (SSD): These constitute a record of changes that have made, together with diagnostics and warnings. They present a record of changes for error correction and recovery.
>
> Reference Reports (REF): These present the information in the data base in various formats. For example, all properties and relationships for a particular object can be shown.
>
> Summary Reports (SUM): These present collections of information in summary form, or gathered from several different relationships. This includes data flows in graphical form and summaries of object types and usage.

Analysis Reports (ANAL): These provide various types of analysis on the information in the data base. These can be used to detect gaps in the information flow or unused data objects.

The reports available through the use of PSL/PSA are designed for several different users: systems users, analysts, designers and project managers. Most reports can be of help to any of these. FIGURE 5.4 shows the main users of each report [ISD79].

In a study done at The University of Saskatchewan [WIG79] to determine which reports were most valuable to an analyst and most frequently used, the following list was arrived at: Name Selection report, Data Activity Interaction report, Formatted Problem Statement, Name List report, Dictionary report, Structure report.

Name Selection report: This is used to select a list of names from the data base which satisfy a given selction criteria. This report is an aid in obtaining other reports which require input lists of a certain type.

Data Activity Interaction report: This report shows the interaction between data objects (SETS, ENTITIES, etc.) and activities (PROCESS, INTERFACE). It also shows data dependancy and interaction among PROCESSES. As well as the information presented, an analysis is also performed which shows up inconsistencies such as data not being used by PROCESSES, PROCESSES not using any data and others.

Formatted Problem Statement: This report presents all information about the objects stored in the data base, and is therefore beneficial in checking the description for accuracy. The Formatted Problem Statement can be maintained as a reference and updated when changes are made to the database.

Name List report: This report presents a list of all objects defined in the data base, along with their type and synonyms. It can be used as a reference to look up names defined in the data base.

Dictionary report: This presents a summary of information for an object or lists of objects. It is valuable for maintaining definitions for names in the data base.

Structure report: This report presents in the form of indented lists, hierarchies of a number of objects based on relationships used in their description. It is helpful in determining loops which have occured in the definition.

PSA REPORTS AND THE ASPECTS THEY PRESENT

Report Name	Aspect of System Description Presented
ASSERTION CONSISTENCY	Communication and Analysis Aids
ATTRIBUTES	Communication and Analysis Aids
CONTENTS	Data Structure
CONTENTS ANALYSIS	Data Structure
CONTENTS COMPARISON	Data Structure
DATA ACTIVITY INTERACTION	Data Derivation/System Flow
DATA BASE SUMMARY	Project Management
DICTIONARY	Communication and Analysis Aids
DYNAMIC INTERACTION	System Control and Dynamics
ELEMENT PROCESS ANALYSIS	Data Derivation
ELEMENT PROCESS USAGE	Data Derivation
EXTENDED PICTURE	System Flow, System Structure, Data Derivation, Data Structure
FORMATTED PROBLEM STATEMENT	All Aspects
FREQUENCY	System Size and Volume
FUNCTION-FLOW-DATA-DIAGRAM	Data Derivation
IDENTIFIER ANALYSIS	Data Structure
KEYWORD IN CONTEXT	Communication and Analysis Aids
LAYOUT	Communication and Analysis Aids
LIST CHANGES	Project Manager
NAME LIST	Communication and Analysis Aids
NAME SELECTION	All Aspects
PICTURE	System Structure, System Flow
PROCESS CHAIN	System Control and Analysis
PROCESS SUMMARY	Data Derivation
PUNCHED COMMENT ENTRIES	All Aspects
RELATION STRUCTURE	Data Structure/Data Derivation
RESOURCE CONSUMPTION ANALYSIS	System Size and Volume
SECURITY ANALYSIS	Data Structure
STRUCTURE	All Aspects
SUBSET ANALYSIS	Data Structure
UNIT STRUCTURE	System Size and Volume
UTILIZES ANALYSIS	System Control and Dynamics

FIGURE 5.3

MANNER OF USAGE OF ANALYZER REPORTS

Report	User	Analyst	Designer	Project Manager
ASCO		*		*
ATTR		*	*	
CONT	*	*	*	
CA		*	*	
CC		*	*	
DAI		*	*	
DBS			*	*
DICT	*	*		*
DI		*	*	
EPA		*		
EPU		*		
EP	*	*	*	
FFDD	*	*	*	
FPS	*	*	*	*
FREQ		*	*	
IA		*	*	
KWIC		*		
LC		*		*
LO	*	*	*	
NL		*		
NS		*		*
PICT	*	*	*	
PC	*	*		*
PSUM		*		*
PCOM		*		
RSTR		*	*	
RCA				*
SECA				*
STR	*	*	*	
SSA			*	
US		*		
UTLA		*	*	

FIGURE 5.4

5.4 Advantages

There are a number of reasons which point to the advantages of using PSL/PSA over a manual documentation method. The advantages discussed in this section have been experienced firsthand and also corroborated by others ([WIN79], [TEI76]). The following advantages are the most noteworthy:

1. PSL/PSA is "amethodological". It does not prescribe how one should go about analyzing or designing a system; it merely documents the result. In this way, as will be shown, it is adaptable to any existing of proposed systems development technique.

2. PSL, when used correctly, allows the development of a well-structured and documented systems description which is clear, concise and consistent. The system description is easily understood, in contrast to the reading of narrative system descriptions which vary stylistically and verbally from writer to writer. PSL also encourages functionality in design which leads to reduced maintainance cost [WIN79].

3. PSL/PSA permits the on-going documentation of a system (either existing or proposed) by as many authors as have access to the system on which the PSL/PSA database resides. This can be invaluable for large systems development efforts where the co-ordination of several independent development groups working on the same system is difficult. PSL/PSA has some project management attributes built into it to further aid in the co-ordination task.

This allows for a greater degree of control over the finished system specification.

4. PSL/PSA facilitates easy and complete updating of documentation; the reports representing the documentation are computer produced rather than done by hand so that any change, once communicated to the data-base, can initiate a complete re-documentation of the system if desired. The documentation becomes a by-product of the development effort instead of being a separate effort on its own ([WIN79], p.286).

5. The syntax checking and report generating capabilities of PSA allow for considerable diagnostic power. Missing relations, undefined activities, redundancies of data, unintended process cycling and a number of other design pitfalls are highlighted in the PSA reports. The result of such feedback is less time and effort spent on implementation of the proposed information system [WIN79].

6. The combination of easy data-base modification and query capabilities allow PSL/PSA to respond handily to innumerable "what-if" type of questions which may occur during the design or maintenance of the system.

5.5 Using PSL/PSA to Model Systems

PSL/PSA has been designed as a tool which may be used to develop a system description. This description, or model, is based on one or more person's view of the system. In this way the model is always at least one step removed from the actual system. The accuracy of this model is critical to the development of an information system, because the proposed system is designed according to the present model and its modifications. The development of an appropriate model depends on both the developer's perspective and the perspective of the tool or technique chosen.

The descriptors of systems theory and the development perspectives discussed in chapter 3 were important factors in choosing PSL/PSA as a tool for modeling other techniques. PSL/PSA allows a system to be defined in terms of its boundaries and environment as well as its internal objects, properties, and relationships. In this way an open system view may be developed. This is important if PSL/PSA is to be used to model other techniques.

As PSL/PSA is used to describe and document a system in a generalized (generic) way, it should not matter whether the systems model being described is in the designer's mind or has been defined by a particular methodology. The only requirement for PSL/PSA is that the system be described according to the object types allowed. The ability to determine these object types relies on the creativity of the person developing the model of the system. One has to be able to "see" the system in PSL's terms.

Once the system is defined, it is analyzed through the use of PSA reports.

5.6 Summary

PSL/PSA is intended to be used primarily as a tool with which systems analysts may document a system (existing or proposed) and check it for basic consistency and completeness. Because of PSL/PSA's attempt to remain amethodological, it has obtained a broader systems perspective than most methodologies and techniques can. It is the intent of the next two chapters to show that PSL/PSA is not only useful as a documentation tool but that it can also be applied to a range of methodologies and techniques. This type of application, in turn, shows that PSL/PSA can not only be used to model a system but that it can also model a model of a system.

CHAPTER 6

PSL/PSA APPLICATION TO BIAIT

BIAIT is one of a number of techniques which attempt to model a system in terms of its information handling disciplines (IHD's). Others include IBM's BSP (Business Systems Planning) and BICS (Business Information Control Study) [KER80] methodologies. The BIAIT model is developed as the result of a number of questions which attempt to define the boundary of the organization's information system. The model is then translated into a BICMX (pronounced "bic-mix", more recently re-termed: BIAIT Print). The PSL/PSA application was done on the BICMX matrix representation of the system, which produces some beneficial results.

6.1 The BIAIT Methodology/Technique

BIAIT, an acronym for Business Information Analysis and Integration Technique, was developed by Donald Burnstine [BUR80]. Its main goal is to describe a customer's needs for computer products and services in terms of information handling functions rather than in terms of the products or services made or provided by the customer ([CAR79], p. 5).

The first step of the BIAIT technique is to derive a classification of the organization in terms of the orders it receives for its products or services [BUR80]. This classification is done with seven questions which, as the result of more than twenty man years of research and testing, have been singled out of more than 400 questions as those being necessary and sufficient to classify a business. The seven classification questions are listed below ([BUR80], p. 7):

1. Does your business bill its customers or accept cash?
2. Does your business deliver its product or service in the future or immediately?
3. Does your business create and maintain profiles of its customers' buying behavior or not?
4. Does your business negotiate price or operate on a fixed price basis?
5. Does your business rent or sell its product or service?
6. Does your business perform product recall and update to the product or service it has offered?
7. Does your business make to order or provide from stock the product or service that it supplies?

These questions may be rephrased to apply at the departmental and occupational level, as well as the organizational. The answers to these questions are used to determine the information handling activities associated with the business. In general, a positive response to a question indicates the more complex case.

There are a number of assumptions or ground rules which must be noted when considering the classification questions. An order is a generic concept which may be formal or informal, i.e. it can be a purchase order, request or even a question. The order must be identified and defined as either a thing (machine, chair, report), space (movie seat, truck compartment) or a skill (mental, physical). The questions are asked from the supplier's (i.e. organization's) point of view, and not the customer's. A supplier can receive many orders, i.e. bill some customers and receive cash from others. The final assumption is that all questions are binary and therefore must be answered with a "yes" or a "no" ([CAR79], p. 5). This would require a rewording of the above questions, i.e. question 7 would become "Does your business make to order?".

The next step is to define the Key Business Resources (KBR). Key resources are those things an organization cares about and therefore keeps track of. A list of typical Key Business Resources would include customer accounts, people, dollars, space, supplies and materials, tools and machines, work in progress, and finished goods. It is the premise of BIAIT that each KBR goes through the 11-step generic life cycle in Fig. 6.1. These resources are related to the organization ([BUR80], p. 14):

> All resources are owned by organization functions which are in turn responsible for taking them through the various steps of their life...Each of these organization functions have been defined in detail according to their planning, control and operations segments. The **planning** segment establishes requirements, long-term commitments and provides estimates. The **control** function allocates resources, evaluates performance, redirects processes, and executes evaluative decisions. The **operations** segment executes routinized procedures and makes rule driven decisions.

The result of determining the Key Business Resources and

LIFE CYCLE NODE	ACTIVITY DESCRIPTION	OBJECTIVES
SET REQUIREMENTS	Project future needs of/for the resource. Problem statements generated here.	Reduce the risk of being surprised by unexpected events internal or external.
SPECIFY	Specify the nature of the resource to be received or supplied.	Reduce the risk of procuring or producing incorrect and/or resource wasting resources.
SELECT SOURCE	Find the most appropriate source(s) of/for the resource.	Reduce the risk of being stuck with one and/or the wrong vendors or customers.
ORDER	Authorize resources to be expended to initiate or procure the resource.	Reduce the risk of getting or doing the wrong things.
ACCEPT ?	Insure that the new resource is acceptable before it moves into or out of the business.	Reduce the risk of getting or supplying bad ones.
INTEGRATE	Add the accepted resource to the present set of items in the inventory of resources.	Reduce the risk of losing it or not being able to find it.
MONITOR PERFORMANCE	Make sure the resource remains acceptable throughout its life.	Reduce the risk of it going sour and no one finding it.
UPGRADE *	Upgrade the resource performance if requirements change.	Reduce the risk of not having the ability to perform future tasks.
MAINTAIN *	Keep the resource in good operating condition.	Reduce the risk of not having it available due to malfunction.
TRANSFER	Move the resource as required.	Reduce the risk of keeping it around too long.
ACCOUNT FOR/ PAY FOR	Keep track of where and how money is spent on or for the resource.	Reduce the risk of expending funds improperly.

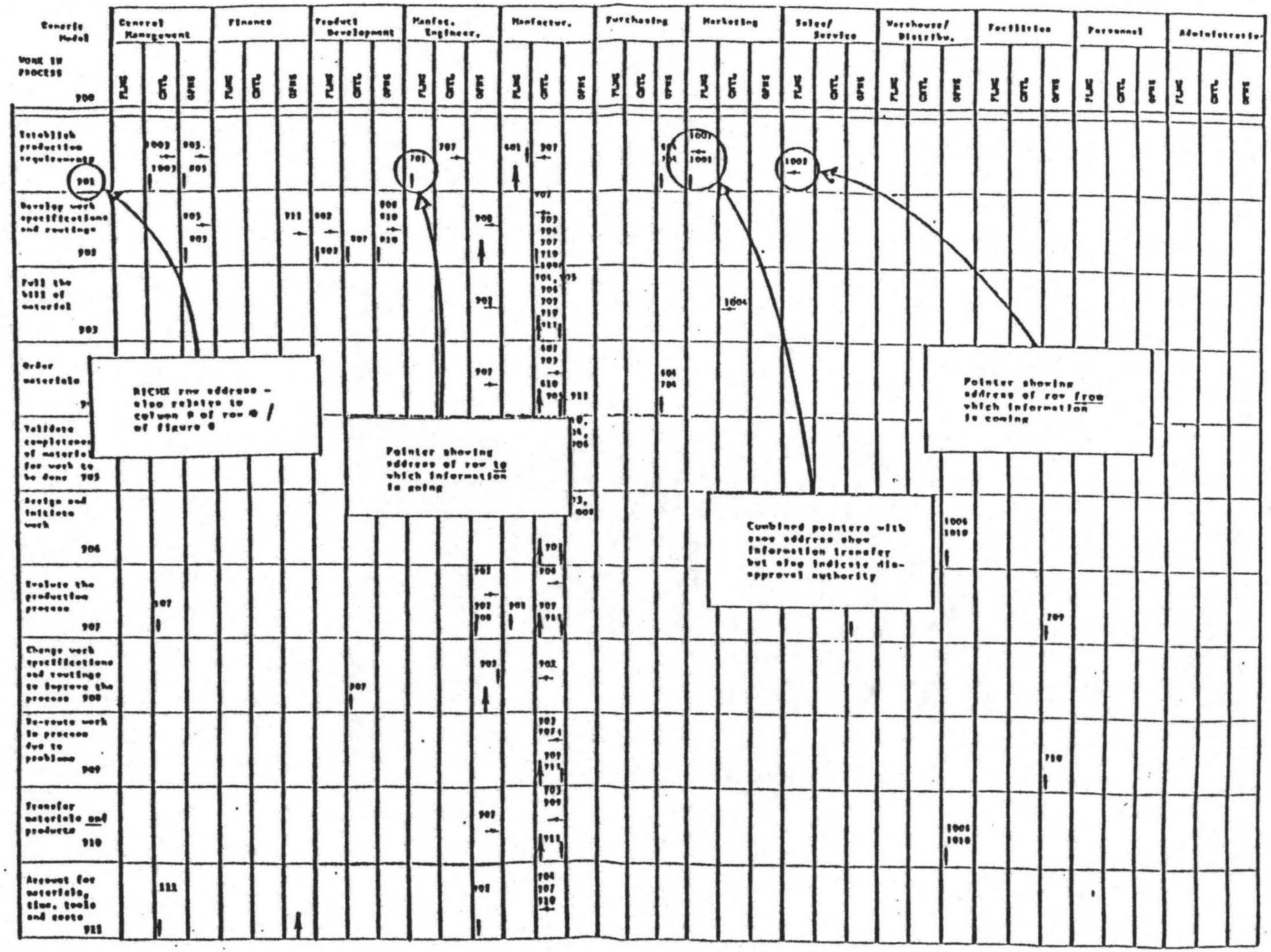
KEY BUSINESS RESOURCE LIFE CYCLE

FIGURE 6.1

defining the organization function is a matrix known as a BICMX (Business Information Control Matrix). The matrix contains the organization functions along the top and KBR life cycle activities down the side (see Fig. 6.2). The set of BICMX's (one for each KBR) is the blueprint for the model of the existing system. In order to complete the blueprint, the relevant matrix cells must be filled in with information interfaces, represented by arrows.

There are four possible information interfaces - responsibility (supplier), user (consumer), influence (input) and authority (dis/approval). The first step in completing the BICMX is to decide where the responsibility (up) arrows go, i.e. which organization function is responsible for which (KBR) activity. There may only be one organization function reponsible for each activity. In addition to responsibility, the BICMX shows where information to perform a given activity comes from [the horizontal (input, influence) arrows] and where the results are sent [the down (user, consumer) arrows] ([BUR80], p. 16). The combination of horizontal and downward pointing arrows is used to show which function has approval and disapproval authority. Along with each horizontal and down arrow there is a number indicating the KBR activity from which information has come, or is going to, respectively.

Once this model of the existing system is derived, it is checked for accuracy. If the model is inaccurate, owing to



SAMPLE BICMX

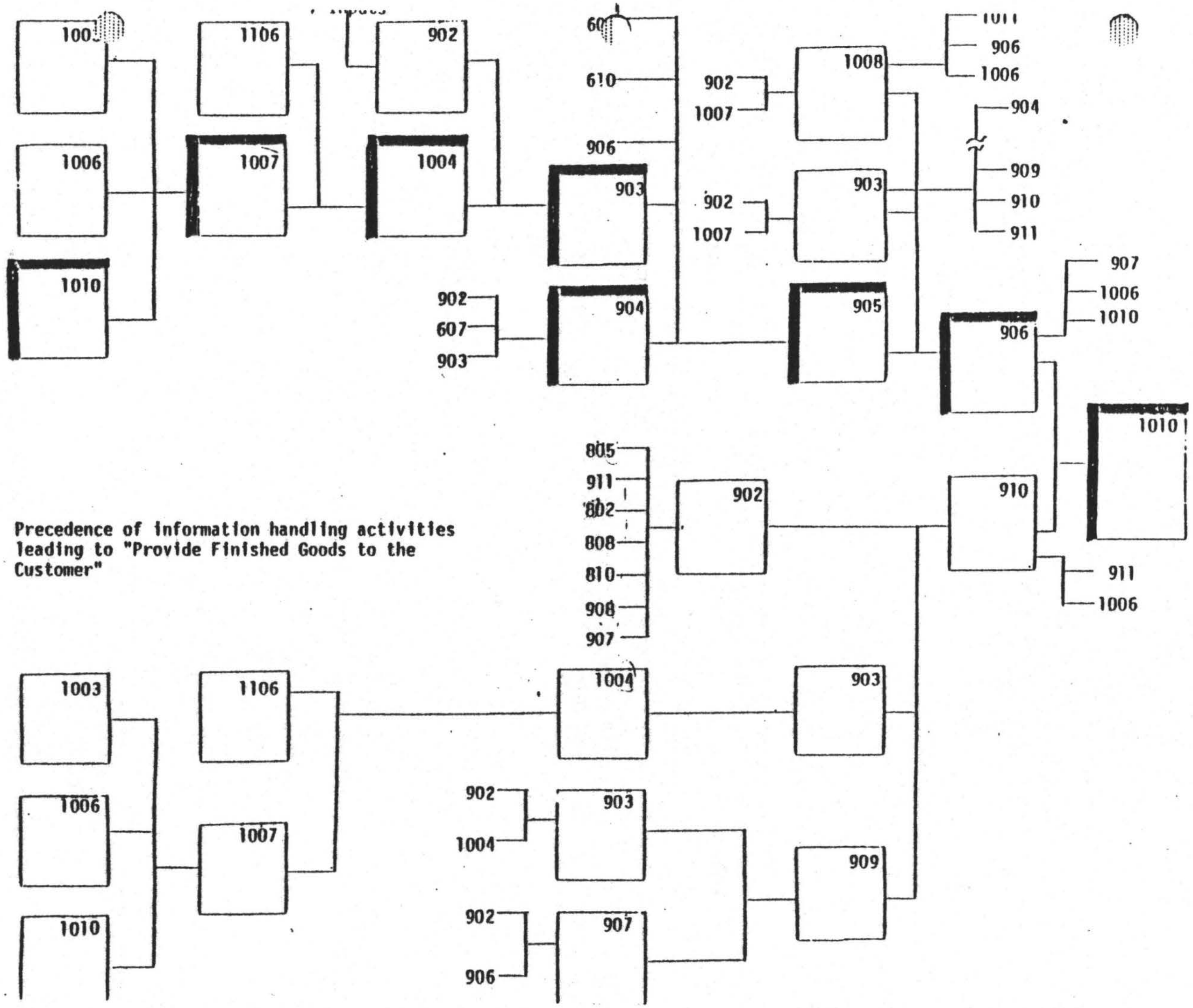
FIGURE 6.2

inaccurate answers to the seven questions, the process is repeated with new answers; otherwise discussion of the model of the new or target system may begin. One of the main goals of BIAIT is that high level management and data processing people may analyze a business information system with a common vocabulary before moving to a discussion of hardware and software needs.

The application of PSL/PSA to the BIAIT BICMX was undertaken for two reasons. One was to determine the utility of the BIAIT to PSL/PSA transformation, [by using a partially completed example handed out as part of an Art Benjamin Associates (ABA) seminar on BIAIT and included as Appendix I] with the primary goal of replicating the hand-generated results appearing in the associated "Precedence of Information Handling Activities" chart (see Fig 6.3). The second reason was an attempt to use PSA reports to explore the BIAIT BICMX model of a system. The BICMX information underlying the chart was entered into the BIAIT.DB PSL/PSA database according to the procedures given in Appendix II. Once entered, several PSA reports (as described above) were produced to verify the data that was entered.

6.2 The Mapping of PSL to the BIAIT BICMX

The BICMX matrix highlights two classes of entities: the BIAIT resource life cycle activities (represented by the rows), and the presumed organizational functions (represented by the



Precedence of information handling activities leading to "Provide Finished Goods to the Customer"

FIGURE 6.3

columns). The cells of the BICMX matrix represent the relations said to exist between the activities and the functions as:

Responsible-for (the up arrows)
Uses-the-results-from/Influenced-by (the horizontal arrows)
Produces/Affects (the down arrows)
Has-authority-over (combination horizontal and down arrows)

The information flows between the activities are implied by the arrows. The actual data, however, which pass between activities are not explicit entities of the BICMX matrix itself. To derive the precedence network of activities, it is necessary to make explicit this "connective tissue" of implied information flow. Thus, the major step necessary to use PSL/PSA to automatically derive the "Precedence of Information Handling Activities" chart, is to define the equivalences between PSL/PSA and BIAIT BICMX.

The BIAIT activities (KBR cells) become PSL/PSA PROCESSES, and the arrows in the BICMX matrix become PSL/PSA ENTITIES representing explicit data messages:

BIAIT Activities (row names) --------------> PSL Processes
BIAIT Implied Information Flows (arrows) --> PSL (Data) Entities

The set of three BICMX's used for the application can be seen in Appendix I.

6.2.1 Entity Definition

As noted above, BIAIT/BICMX does not formally name the information which is being passed from one BICMX cell to another.

It is therefore necessary to formulate a simple but effective naming convention for the flows implied by the arrows in the associated BICMX matrix. For each BICMX row there is one "output" entity, which is associated with the flow designated by the up (responsibility) [↑] arrow. The name given to the output entity for that row is formed by taking the noun form of the name of the row.

Using the first BICMX example of Appendix I (see Fig. 6.4), the noun from the row 801 (forecast market demand) yields the output entity name: "market-demand-forecast". It is convenient to have a "short-name" for this entity for purposes of simplified print-out, updating and direct equivalence with the BICMX matrix. For this example the synonym will be E801 (the Entity representing the upward pointing arrow in row 801).

For each downward pointing arrow in each BICMX row there will be a PSL/PSA output entity sub-part associated with the output entity (as discussed above). The concept of a "sub-part" can be thought of as a "copy" of the output entity derived by the row activity, akin to (say) part 2 of a 3-part form. The sub-part "short-name" will take the form:

SPsss-ddd where "sss" represents the row number of the source activity creating the output entity and "ddd" represents the destination row (activity/process) to which the output is directed by this arrow.

To continue the example for row 800, the downward pointing

Model	Management			[illegible]			Development			Engineer.			[illegible]			[illegible]			[illegible]			Service			Distribu.			Facilities			Personnel			Administ	
PRODUCT DEFINITION 800	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL
Forecast market demand for product function 801	101 1101 →	803	805 → 805				802 804 → 802		808 → 808										203				207 →												
Specify particular product characteristics and quantities 802	101 →	102 → 803	805 → 805			811	806 ← 804			701		902 ← 902							801 ← 801																
Select a development site 803					302 ←		802 ← 807 804												801 ←									501		508					
Schedule development 804		803 →	104 → 106			811	802 ← 802												801 ←																
Review development activities 805		107 ←			307 ←		802 ← 802	806	808 ← 808			902 ← 902	901 ← 901						801 1002 ← 801 1002																
Add accepted specs to the inventory of specifications 806			805 ←			811																													
Evaluate specifications usage and performance 807		107 1003 → 1003			307 →	811			808			902 908 ←											207 ←												
Upgrade product specification as necessary 808			108 109 803 → 803			811		807 →		701		902							801 ← 801																
809																																			
Transfer product specs to manufacturing 810		107 110	106			811						902 ← 902																							
Account for development costs 811					307		802 804 ←	806 807 ←	808 810 ←																										

BIAIT BICMX Example

FIGURE 6.4

arrow directed to cell 805 would be coded as "SP801-805" [which translates as: "the sub-part of E801 which is sent to (received by) process A805"].

The horizontal arrows [<-,->] represent output sub-parts coming from other activities and appearing as input sub-parts to this activity. Because of this two-way relation (one activity's output sub-part is another activity's input sub-part), a labelling convention for sub-parts had to be chosen, which was the same, irrespective of whether we are looking at its source activity or its destination activity. We arbitrarily adopted the labelling convention of the output side. Thus a sub-part coming into (say) activity 801 from 806 would be labelled SP806-801. As will be seen such a consistent labelling convention permits a very simple diagnostic check for missing inputs/outputs in the BICMX matrix.

6.2.2 Process Definition

For each row of the BICMX there will be a PSL/PSA PROCESS (dually) defined by both the activity number and its name. The "short-name" will be the activity row number preceded by an "A" (e.g. A801). The longer description of the process will be the name of the activity (e.g. forecast-market-demand).

6.2.3 Entity-Process Relation Definitions

To inter-connect the processes (as required to derive the sought after process network diagram - FIG. 6.3), it is necessary to describe to PSL/PSA the relations between processes, in terms of their use of the intermediary flows of information. These inter-process information flows are the entity sub-parts described above. This connection is made directly from the BICMX matrix by focusing on the downward pointing and horizontal arrows in a given row. Using the terminology of PSL/PSA, each PROCESS (row) will be said to EMPLOY the input enitity sub-parts [i.e. any sub-part with horizontal (<-,->) arrows], and will DERIVE the output entity sub-parts (i.e., any sub-parts with downward pointing arrows).

6.2.4 Summary Example

Focusing on activity 801 from the example of Appendix I, the transformation into exact PSL syntax (suitable for entry into the PSL program of PSL/PSA) would be as follows:

```
DEFINE ENTITY                                   E801;
     SYNONYM                                    MARKET-DEMAND-FORECAST;
     SUBPARTS ARE                               SP801-802,
                                                SP801-803,
                                                SP801-805,
                                                SP801-808;

DEFINE PROCESS                                  A801;
     SYNONYM                                    FORECAST-MARKET-DEMAND;
     DESC;
        This process forecasts the market demand for the product
        function;
```

EMPLOYS	SP101-801,
	SP202-801,
	SP207-801,
	SP802-801,
	SP804-801,
	SP805-801,
	SP808-801,
	SP1101-801;
DERIVES	SP801-802,
	SP801-803,
	SP801-805,
	SP801-805;

6.3 PSA ANALYSIS OF THE BIAIT EXAMPLE

In order to analyze the PSL description of the BIAIT BICMX, a number of PSA reports were generated. Three of these PSA reports (see Appendix III) were seen to be immediately beneficial in terms of BICMX analysis. They are: the Formatted Problem Statement (FPS), the Data Activity Interaction (DAI) report, and the Extended Picture (EP) report. The first two were found useful for verification of the model and the third was able to duplicate the "Precedence of Information Handling Activity" graph.

The FPS report identified entity sub-parts which were "employed" by a process but were not defined by the process generating that sub-part. In terms of the BICMX notation this would translate to the case where, in one row, a horizontal arrow indicates the input of an information flow from another row, but there is no corresponding downward pointing arrow in the source row. There were several such omissions, which upon further checking, appear to be errors in the ABA example, not in the data entry. The specific errors uncovered are reported in Fig. 6.5.

The DAI report, through its matrix analysis, diagnoses the reverse case, i.e., when a row has a downward pointing arrow indicating a flow to another row, but the corresponding row has no horizontal arrow indicating its receipt. Here again, several inconsistencies were found in the ABA example. The correction procedure in both cases was to enter (via PSL) the missing definitions, and to then re-run the reports to make sure that all of the inconsistencies had been found. The details are in Fig. 6.5.

Once the above edit/verification process is completed, the Extended Picture (EP) report can be used to generate the precedence network diagrams. As noted in the preceding section, there are several variations of this report. Working either forwards or backwards from a specfic BIAIT activity (PSL/PSA PROCESS) one can elect to show or supress the intervening information flows (represented as separate entity sub-part boxes). Here again, differences were found between the diagram generated for the ABA example and the diagram produced by PSL/PSA's EP report. Upon investigation this was attributable to potential omissions in the ABA example diagram. The results of the EP report for the example are found in Appendix IV.

Even if the PSL/PSA analysis were only used for checking

SPECIFIC INCONSISTENCIES FOUND IN THE BICMX EXAMPLE

Inconsistencies found using FPS

ACTIVITY	RECEIVES FROM ACTIVITY	BUT	DOESN'T SEND IT
804	801	801	
1002	802	802	
1001	802	802	
801	804	804	
1005	902	902	
905	906	906	
910	909	909	

NB: The last "error" may be a compound error involving two missing flows; it is assumed that the flow to 911 should read a flow from 910.

Inconsistencies found using DAI

ACTIVITY	SENDS TO	BUT	DOESN'T USE IT
1002	1003	1003	
803	802	802	
903	911	911	
909	911	911	

NB: The last "error" may be a compound error as noted above.

Inconsistencies found using EP

While they cannot be called inconsistencies per se, the network diagram drawn in the example contains missing chains which are evident in the diagram drawn by the EP report.

FIGURE 6.5

and reporting of the BICMX matrix and the production of its associated diagrams, it is clear that it is capable of showing up errors and inconsistencies that could be difficult or impossible to detect by human means and would alone justify its use. However, there are many other applications of PSL/PSA to BIAIT and a few of these are taken up in chapter 8.

CHAPTER 7

PSL/PSA APPLICATION TO STRUCTURED ANALYSIS

A number of methodologies have been developed recently which all fall into the category of structured analysis. Most of them attempt to cover the life cycle from the inquiry into user requirements to the target specification. The one chosen for this application is that of Tom DeMarco [DEM78] of YOURDON inc., but there many similiar techniques, such as Softech's SADT [ROS76], Gane and Sarson's "Structured Systems Analysis" [GAN79], and EXXON's SSA [MEN80].

The application of PSL/PSA to Structured Analysis is examined with respect to two specific areas. First, how applicable is PSL in terms of describing a system which has been represented using Structured Analysis? Second, to what extent are the report generating capabilities of PSA of use as an aid to this method of system analysis?

7.1 Structured Analysis

The Structured Analysis technique of Demarco begins by documenting the existing information system and ends with a target document or systems specification of the new information system.

To arrive at the systems specification (analysis), the use of a number of tools is suggested. These tools are Data Flow Diagrams, Data Dictionaries, Data Structure Diagrams, Structured English, Decision Tables and Decision Trees.

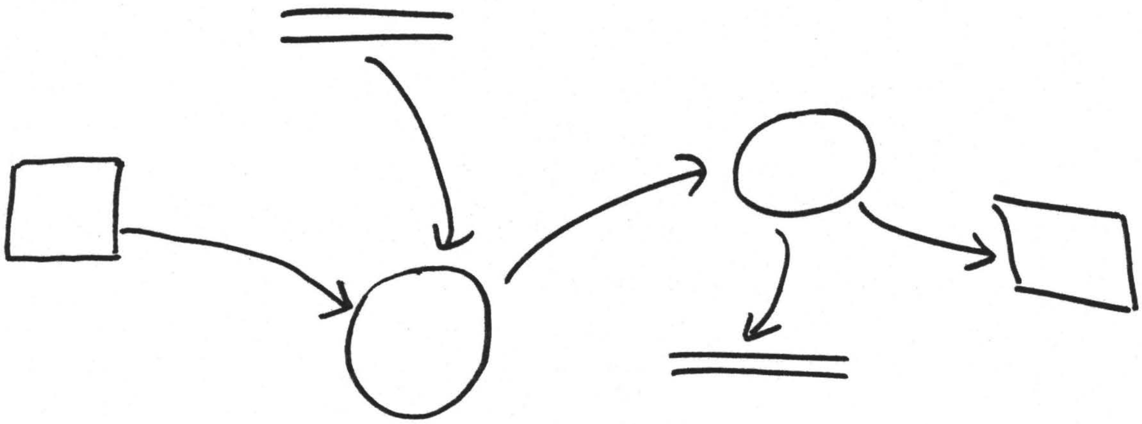
One of the major objectives of Structured Analysis is to arrive at an accurate and useful representation of the system without writing a "monolithic Victorian Novel". To achieve this end the system is partitioned into a number of mini-specifications. The major partitioning tool is the Data Flow Diagram (DFD) which DeMarco defines as follows ([DEM78], p. 47):

> A **Data Flow Diagram** is a network representation of a system. The system may be automated, manual or mixed. The Data Flow Diagram portrays the system in terms of its component pieces, with all interfaces among the components indicated.

There are four basic elements which make up a Data Flow Diagram. They are [DEM78]:

> **Data Flows** (represented by named vectors/directed arcs): A data flow is a pipeline through which pockets of information flow.
>
> **Processes** (represented by circles): A process is a transformation of incoming data flow(s) into outgoing data flow(s).
>
> **Files** (represented by stright lines): A file is a temporary repository of data.
>
> **Data Sources and Sinks** (represented by straight lines): A source or sink is a person or organization, lying outside the context of a system, that is a net originator or receiver of system data.

The primary purpose of DFD's is to represent the system from the point of view of the data, as opposed to flowcharts which show it from the point of view of those processes which act upon the data. The aim is to go beyond the datalogical (flow of control) perspective and achieve a higher level (flow of data) model of the system. Figure 7.1 shows an example Data Flow Diagram.



DATA FLOW DIAGRAM
FIGURE 7.1

A particular target system's DFD is the result of interviews, observations and analysis of reports which flow through the system. A complete view of the system will result in a set of leveled Data Flow Diagrams. A top-down approach is taken, with the top (Context) diagram showing the system as one process with a number of inputs, outputs, sources and sinks. This view of the system is then partitioned into a number of subsystems (or processes). Each of these are in turn partitioned until the bottom level of functional primitives is reached (see Appendix VI). A standard numbering convention is followed in order to keep track of levels and processes.

The second tool suggested by DeMarco is a Data Dictionary (DD). There are four classes of items in a Data Dictionary. They are data flow, file, process, and data element. The first three are components of the DFD's and a data element is a data flow which cannot be decomposed. Figure 7.2 shows a typical Data Dictionary entry for a data flow. File and data element entries are similiar.

```
Order    =  * tentative offer for goods *
            [Buyer-name | Buyer-ID] + 1{Qty + Item-description}
```

DATA DICTIONARY ENTRY
FIGURE 7.2

Process entries, on the other hand, make use of the third set of tools: Structured English (SE), Decision Trees and Decision Tables. These may be grouped under the heading of mini-specifications. All process entries in a DD are functional primitives, i.e. they cannot be partitioned any further. Each entry contains the process name, number and description in the form of a mini-specification.

Once the existing system has been modeled using the tools of Structured Analysis, the Structured Specification is developed through the logical generalization of DFD's, establishing the user requirements and the Document of Change, generating alternatives and selecting an option; in other words, acting out the rest of a typical life cycle.

The case study used for this application of PSL/PSA is one taken from the Structured Analysis seminars given by YOURDON inc [YOU79]. An overview of the case, based on the "Fromage du Midi" (part of a wholesale market located in France), is given in Appendix V. Included in the case study are a summary of the operation, two preliminary Data Flow Diagrams, the reason for requesting a study, a record of interviews with the three main employees and a Data Dictionary.

In approaching this case, the first step taken was to apply Structured Analysis in order to derive the necessary Data Flow Diagrams. These can be seen in Appendix VI. The DFD's were then translated into PSL using an OPR (Object-Property-Relation) map and entered into the PSA analyzer data base. The third step was to generate a number of PSA reports and examine them.

7.2 Applying PSL to the Tools of Structured Analysis

In order to describe an existing system using Structured Analysis, DeMarco proposes the use of three tools. The tools are Data Flow Diagrams, a Data Dictionary, and mini-specifications in the form of Structured English, Decision Trees and/or Decision Tables. Each of these tools has, in turn, been translated into PSL.

Data Flow Diagrams are used to arrive at a graphical representation of the system. The system is defined in terms of its data flows, files, and processes. Any set of DFD's contains three distinct level groupings. The top level is a Context Diagram consisting of a single process bubble, a number of boxes representing data sources and sinks, and a number of data flow vectors which show information entering or leaving the target system. The bottom level diagrams consist of process bubbles which cannot be broken down any further. These are known as functional primitives. The intermediate level(s), all those between the top and bottom, consist of a hierarchy of processes which are partitioned into successively finer and finer groups until the functional primitives are reached.

In order to distinguish among the three groups of DFD levels, different PSL verbs were used for each one. The Context Diagram is described in terms of INPUTS to and OUTPUTS from the target system. The data sources and sinks become INTERFACES and together with the top level PROCESS they RECEIVE or GENERATE all data flows. On the functional primitive level (see EX. 1) all PROCESSES are described in as detailed a manner as possible, using three and/or four part relationships (DERIVES/USING, ADDS/MODIFIES/REMOVES, etc.). The intermediate levels (see EX. 2) are described using simpler (i.e. two part) relationships such as DERIVES, UPDATES and EMPLOYS.

To conform to the DFD numbering convention, two ATTRIBUTES have been defined (see EX. 3). They are Diagram and Process-Number. Diagram is given a value in the form of "D-"n, where n is the diagram number, i.e. 1 or 2.1.3. Process-Number takes on a value in the format "P-"m, where m is the process number (integer) on that particular DFD. This numbering convention allows the selection of any particular PROCESS in the system or of all PROCESSES contained on a DFD page. All PROCESSES which result from the partitioning of a specific process bubble are designated as its SUBPARTS (see EX. 3).

Included in the case study is a partial Data Dictionary. A Data Dictionary is the result of a collection of documents, reports and other types of information which flow through the system. There are four classes of components in a Data Dictionary: data flow, file, process and data element. The corresponding PSL object types are ENTITY, INPUT, OUTPUT, SET, PROCESS and ELEMENT. DeMarco's Data Dictionary entries are defined using a set of five relational operators. These operators and their symbol notation are: IS EQUIVALENT TO - '='; AND - '+'; EITHER-OR - '[]'; ITERATIONS OF - '{}'; OPTIONAL - '()'. In addition, comments can be added in the form of *...text...*. An example of the Data Dictionary entry for Order-Form follows:

```
Order-Form      =  *documentation of a real or tentative sale*
                    Wholesaler-ID + [Buyer-ID|Buyer-Name] +
                    1{Qty + Item-Desdcription + Amount} +
                    Total-Amount
```

The PSL relationships and system-parameters corresponding to

EX. 1.
PSL DEFINITION OF A FUNCTIONAL PRIMITIVE

```
DEFINE PROCESS          Select-and-Regroup-Items;
     DERIVES               Filled-in-Ticket
       USING               Ticket-Copy;
     DERIVES               Check-Out
       USING               Credited-Order-Form;
     MODIFIES              Hanger
       IN                  Sales-Area;
     MODIFIES              Arrival-Receipt
       IN                  Receipts-File;
     REMOVES               Ticket-Copy
       FROM                Sales-Area;
```

EX. 2.
PSL DEFINITION OF AN INTERMEDIATE LEVEL PROCESS

```
DEFINE PROCESS             Selling-Goods;
     DERIVES               Order-Packet, Credit-Request,
                            Sales-Receipt;
     UPDATES               Receipts-File, Sales-Area;
     EMPLOYS               Order, Confirmation;
```

EX. 3.
ADDITIONAL PSL PROCESS DESCRIPTIONS

```
DEFINE PROCESS          Selling-Goods;
     ATTRIBUTES ARE        Diagram        D-0,
                           Process-Number P-1;
     SUBPARTS ARE          Document-Tentative-Order,
                           Check-Credit-Hold-Funds,
                           Select-and-Regroup-Items,
                           Revise-Order;
```

PSL DATA FLOW DIAGRAM DEFINITIONS
FIGURE 7.3

DeMarco's relational operators are seen in Table 1. The entries in this part of the Data Dictionary are data flows and files. They are defined as ENTITIES, INPUTS, OUTPUTS and SETS and the relational operators are generally applied to GROUPS and ELEMENTS.

The PSL definition needed for the Data Dictionary entry Order-Form is shown in EX. 4. Each non-divisible data item (ELEMENT) is also given a description (see EX. 5).

For each functional primitive there must be a mini-specification which describes how the process is carried out. These mini-specs (Structured English, Decision Tables and Decision Trees) may be defined either under the relationship heading Description or Procedure (see EX. 6). Both relationships are for comment entries so the mini-spec may be written out in whatever format is applicable.

The transformation of the three tools of Structured Analysis into PSL was accomplished with varying degrees of compromise. The activity bubbles on the Data Flow Diagrams were defined as the PSL object type PROCESS and then described in terms of the operations they performed on the data flows entering and leaving. There are two object types which can be used for defining the internal data flows. They are GROUP and ENTITY. Flows entering or leaving the target system are defined as INPUTS and OUTPUTS. Their properties are similiar to those of ENTITIES. The

TABLE I

YOURDON -> PSL/PSA DATA DICTIONARY CONVERSION

SYMBOL	RELATION	PSL EQUIVALENT	PSL TYPE
=	IS EQUIVALENT TO	CONSISTS OF(CSTS)/ COLLECTION OF(CLTN)	Relationship
+	AND	implied for objects related to a CSTS or CLTN statement	-
[]	EITHER OR	ONE-OF-THE-FOLLOWING	SYS-PAR
{}	ITERATIONS OF	ONE-OR-MORE/ ONE-TO-TEN,etc.	SYS-PAR
()	OPTIONAL	OPTIONAL	SYS-PAR

FIGURE 7.4

EX. 4.
PSL DEFINITION FOR A DATA DICTIONARY ENTRY

```
DEFINE ENTITY          Order-Form;
     DESCRIPTION;
       This is the documentation for a real or tentative sale.;
     CONSISTS OF          Order-Form-Group;

DEFINE GROUP           Order-Form-Group;
     CONSISTS OF          Blank-Order-Form,
                          Wholesaler-ID,
                          Amount-of-Sale,
                          One-or-more             Item-and-Price,
                          One-of-the-following    Identifications;

DEFINE GROUP           Item-and-Price;
     SYNONYMS ARE         Items-and-Price;
     CONSISTS OF          Quantity, Item-Description;

DEFINE GROUP           Identifications;
     CONSISTS OF          Buyer-Name, Buyer-ID;

DEFINE ELEMENT         Blank-Order-Form, Wholesaler-ID,
                       Amount-of-Sale, Quantity,
                       Item-Description, Buyer-Name,
                       Buyer-ID;
```

EX. 5.
PSL DEFINITION OF A DATA ELEMENT

```
DEFINE ELEMENT           Quantity;
     SYNONYMS ARE         Qty;
     DESCRIPTION;
       This indicates how much of each item was purchased.;
```

EX. 6.
PSL DEFINITION OF A STRUCTURED ENGLISH MINI-SPECIFICATION

```
DEFINE PROCESS           Select-and-Regroup-Items;
     PROCEDURE;
       For each Buyer:
         1. Guide them through the Sales Area and negotiate price.
         2. If Buyer selects goods
              Remove Ticket from Hanger
              Mark condition of sale on top copy
              Initial Buyer's Order Form to signify a sale
              Send Filled-in-Ticket and signed Order Form to
                Manager to finalize sale
            Otherwise
              Mark Buyer's Order Form 'No Sale'.
         3. If part of a Hanger is sold
              Split Hanger into sold and unsold parts
              Revise the Hanger Ticket
              Update the Receipts File.;
```

PSL DATA DICTIONARY EXAMPLES
FIGURE 7.5

major distinquishing factors (in this case) between ENTITIES and GROUPS are their data structure and the relationships they can be part of.

The data structure of both types is made up of a hierarchy of GROUPS with ELEMENTS (non-divisible data item) at the lowest level. The difference lies in the fact that two GROUPS may be combined into a higher level GROUP which CONSISTS OF both lower level GROUPS. An ENTITY, however, may not be combined with another ENTITY or GROUP to form a higher level ENTITY. The data derivation statements (DERIVES/USING, etc.) in a PROCESS allow an ENTITY and a GROUP to combine to form an ENTITY, but the resulting ENTITY may not CONSIST OF anything but GROUPS and ELEMENTS. The use of GROUPS as the object types for data flows would cause less redundancy in conforming to the style of DeMarco's Data Dictionary.

Although there are some apparent advantages to defining data flows as GROUPS, the choice was made to define them as ENTITIES for a number of reasons. First, even if the internal data flows are defined as GROUPS, the INPUTS and OUTPUTS cannot be redefined as GROUPS and so the same problem of combining INPUTS and GROUPS to form other GROUPS still exists. Also, the only data object types allowed in four part relationships (ADDS/REMOVES/MODIFIES/etc) are ENTITIES. To eliminate these relationships results in a weaker system description. The problem of combining ENTITIES and maintaining internal data structure can be solved by associating an all-encompassing GROUP (i.e. a GROUP which CONSISTS OF all GROUPS and ELEMENTS within the ENTITY) with an ENTITY when it enters a PROCESS. This GROUP may become, as applicable, part of a data flow (ENTITIY) leaving the PROCESS.

The result of defining data flows as INPUTS, OUTPUTS and ENTITIES is that a consistent system description is maintained. The only disadvantage appears to be the padding of data structure definitions with a number of intermediate GROUPS.

Once the data flow object types were decided, the translation of Data Dictionary entries into PSL was quite straightforward. Each data flow was defined as an INPUT, OUTPUT or ENTITY CONSISTing of a hierarchy of GROUPS and ELEMENTS, according to the actual reports and the data flow conventions described above. The PSL system description includes a comment entry for each data flow, file, and data element, and a mini specification of the procedure of each functional primitive is included in a Data Dictionary.

7.3 PSA Analysis

Many of the available PSA reports were useful to this study. Their usefulness can be divided into three categories: reports which mimic the tools of Structured Analysis; reports which aid in the diagnosis of the system description; and reports which have not been examined with regard to their value in this type of systems analysis. Appendix VII contains a sample of each PSA report mentioned in this section.

Both the Picture Report and the Extended Picture Report, subject to the hierarchical limitations, can be used to derive graphs similiar to the Data Flow Diagrams. The Context Diagram is derived by submitting the top level PROCESS name to the Picture Report. If the sequence of functional primitives is linear, or at least has a definite starting point, the Extended Picture Report may be used to lay out a full view of the system. The Function Flow Data Diagram and the Picture Report are both able to give a snapshot view of any process, showing all its entry and exit data flows. The Formatted Problem Statement gives a complete view of the system, in list form, according to object type.

The closest report to a Data Flow Diagram is the Extended Picture Report. The Extended Picture may be used to derive a DFD using any process, data flow, or data source or sink name as a starting point. The picture may be generated in either the forward or backward direction. A picture generated in the forward direction would start with an INTERFACE or INPUT and show the chain of processes and data interactions which occur as a result of its association with the system. Specifying a backward direction and using an OUTPUT of INTERFACE as a starting point, would result in a picture showing all interaction which lead up to the OUTPUT or INTERFACE specified. For a complete overview it is best to use an INTERFACE (data source or sink) name or the name of a PROCESS which receives INPUT or generates OUTPUT.

There is one problem with the Extended Picture (EP) Report, in terms of resembling a DFD. The problem is that the report does not distinquish between levels of processes, and so data flows which appear on more than just the lowest level DFD are shown. This results in chains of processes which are not functional primitives appearing on the EP report. One possible solution might be to add a suffix to each of the data flows designating its level or diagram number.

A Data Dictionary, as envisioned by DeMarco, must include an entry for each data flow, file, process and data element. It can be made up as a combination of three PSA reports (see EX. 7). For each entry in the Data Dictionary a description is needed. These descriptions are defined under the PSL relation DESCRIPTION. They may be retrieved using the Dictionary Report, which also includes any SYNONYMS and ATTRIBUTES the object type has. A DD is also to include a structured decomposition of each data flow and file. This feature is available in two reports, the Structure Report and the Contents Report. The Contents Report suits the needs of the DD a little better because it includes the relational operators in the form of SYSTEM-PARAMETERS. Because of the consistency in the DFD to PSL transformation, only four object types need to be examined via the Contents Report. They are SET, ENTITY, INPUT and OUTPUT. The Structured English description of each functional primitive can be obtained in any one of three reports. If the Structured English is listed under DESCRIPTION, it can be part of the Dictionary Report. If, on the other hand, it is included under PROCEDURE, it may be derived by submitting the list of functional primitives to either the Process Summary or the Punched Comment Entry Report. The Process Summary includes a list of the data entering and leaving the process as well.

Aside from the reports which mimic the tools of Stuctured Analysis, there are other PSA reports which may be of aid to the analyst. The Name List Report can be used along with the Data Dictionary reports as an index for quick lookups of names which are in the DD. The Data Base Summary can be used to determine if all of the data flows, files, and elements have DESCRIPTIONS, ATTRIBUTES or other relationships defined for them. The Data Activity Interaction Report has useful diagnostic capabilities, which may be of use in the Structured Analysis phase of the life cycle. These capabilities include checking whether data items (ENTITY, SET, GROUP, ELEMENT) are derived by a process, whether they are used once they are derived, and whether processes use or derive and data. Both the Data Activity Interaction (DAI) Matrix Analysis and the Process Interaction Analysis help to point out inconsistencies in the system description.

THREE PSA REPORTS WHICH ARE USED FOR DATA DICTIONARY ENTRIES

Contents Report

```
1* (ENTITY)            1 Order-Form
  1 (GROUP)              2 Order-Form-Group
     2 (ELEMENT)             3 Blank-Order-Form
     3 (ELEMENT)             3 Wholesaler-ID
     4 (ELEMENT)             3 Amount-of-Sale
     5 (GROUP)               3 Item-and-Price (One-or-more)
     6 (ELEMENT)              4 Quantity
     7 (ELEMENT)              4 Item-Description
     8 (ELEMENT)              4 Amount
     9 (GROUP)               3 Identifications (One-of-the-following)
     10 (ELEMENT)              4 Buyer-Name
     11 (ELEMENT)              4 Buyer-ID
```

Process Summary

```
1*    Select-and-Regroup-Items

      For each buyer:
        1.  Guide through Sales Area and negotiate price
        2.  If buyer selects goods
              Remove Tickets from Hanger
              Mark conditions of sale on top copy
              Initial Buyer's Order Form to signify a sale
              Send Filled-in-Ticket and signed Order Form to
                Manager to finalize sale
              Otherwise
                Mark Buyer's Order Form 'NO SALE'
        3.  If part of a Hanger is sold
              Split Hanger into sold and unsold parts
              Revise the Hanger Ticket
              Update the Receipts File.

  INFORMATION ENTERING
    1 Ticket-Copy              ENTITY          USED TO DERIVE
    2 Credited-Order-Form      ENTITY          USED TO DERIVE

  INFORMATION LEAVING
    1 Filled-in-Tickets        ENTITY          DERIVED
    2 Check-Out                ENTITY          DERIVED
```

Dictionary Report

```
1 Amount-of-Sale                                      ELEMENT
 DESCRIPTION:
   This is the amount the Buyer has spent.
                         SYNONYMS: Amt-of-Sale
                                   Total-Amount
                                   Total-Amt
2 Select-and-Regroup-Items                            PROCESS
                         ATTRIBUTE: Diagram
                             VALUE: D-1
                         ATTRIBUTE: Process-number
                             VALUE: P-3
```

FIGURE 7.6

There are also reports which may be of aid in the post analysis phase of the project life cycle. The aim of Structured Analysis is to produce a Structured Specification. The Structured Specification is composed of the aforementioned tools as well as a logicalized DFD showing the processes which are to be automated, semi-automated or left manual, and a Data Structure Diagram. The Data Structure Diagram is derived through a logical generalization of the files found in the Data Dictionary. This process is explained in chapter 19 of DeMarco's book [DEM78]. There is no report which performs this generalization, but the procedure is structured enough to allow one to be written. The Extended Picture Report allows the system to be seen as a whole and this can help in deciding where to draw the boundaries of automation.

PSA features a number of other reports as well. These are described fully in the PSA reports manual [ISD79b].

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

The initial purpose of this paper was to do a trial application of PSL/PSA in order to examine its abilities in terms of documentation, and applicability to several of system development techniques. A number of conclusions and recommendations may be made about these applications, and about PSL/PSA itself.

8.1 The PSL/PSA - BIAIT Application

The application of PSL/PSA to the BIAIT BICMX produced some significant results. As discussed previously, the verification abilities alone were important enough to warrant further application and research. Upon further application to a complete set of BICMX's (11), the number of descrepancies found increased dramatically (to greater than 180 just by using the FPS report analysis). The production of the precedence chart with the EP report was also important, as it duplicated a chart (Fig. 5.3) which had taken two days to do manually, and even then had an error in it. Also, the EP report can produce a precendence chart for any activity and for any depth required, once the system description is stored.

In addition to the above mentioned reports, which were found immediately useful, a number of other reports were explored. The reports which showed signs of usefulness included the Structure (STR) report, Dynamic Interaction (DI) report, Function Flow Data Diagram (FFDD) and aspects of the EP report not discussed previously. Work has also begun on a possible "ERA"-based analyst support system for such methodologies and techniques as BIAIT [WEL80b].

Another possiblity for further extension of this research is in the area of the PSL description of the BIAIT BICMX. In generating the information precedence matrix, only two entity-types contained in the BICMX were used: data-entities (implicit) and activity entities (the rows). The BICMX column information of organizational functions were not defined in PSL, or used in the generation of any of the reports. There is, however, nothing to prevent their declaration to PSL/PSA as "function entities" as well as their associated relations, both with the information (data entities) and with the activity entities, forming a type of three-dimensional network of relations. The PSL "object" called PROCESSORS could be used directly, or the PSL ENTITY object could be used in two different ways (as data and as function). This might permit a much richer set of relationships to be developed than the arrow conventions currently used in a dual role. PSL/PSA could then be used to "slice" the three dimensional stored image in a number of different and perhaps, diagnostically revealing, two dimensional PSL/PSA reports. Because of the ability to incrementally add information to the PSL/PSA database, it is then possible to focus on a particular subset of relations which, for a given organization, may be most convenient/meaningful to develop first.

It became apparent, while documenting the BICMX example, that very little of the "power" of PSL/PSA was being used in terms of "object" descriptions and attributes. While we have not had time to relate all of these PSL/PSA defined properties to the BICMX description we believe that a number of them may have utility. Little, if anything has been done regarding the data entities themselves, once defined by the PSL description. Using the ability to describe the data entities, one can "pull out" for examination (using PSA reports) the data entities themselves. PSL/PSA provides the inherent power to do this through the higher (but thus far not used) concept of SETS and the ability to sub-part and regroup data entities without affecting the BICMX relations. The results of such an analysis can then be fed back to the PSL/PSA database description as another set of relations to look at in terms of the existing BICMX relations.

8.2 The PSL/PSA - Structured Analysis Application

The application of PSL/PSA to the techniques of Structured Analysis has been successful in both areas of examination. The translation of the system description, arrived at using Structured Analysis, into the object type definitions of PSL, was accomplished with minimal redundancy and no loss of meaning or intent. The reports generated using PSA were useful in their ability to reproduce the tools of Structured Analysis, and as diagnostic aids in terms of the completeness of the Stuctured Analysis phase of the project life cycle.

PSL/PSA is readily adaptable to the use of Structured Analysis, but this is a minor point if there are no advantages to this combination. Fortunately there are a few. A very obvious advantage is the degree of automation afforded through the use of PSL/PSA. This is best seen in the ability of PSA to maintain a current version of the Data Dictionary. As new information is encountered, the data base analyzer checks for consistency. Modifications are made accordingly and new copies of Data Dictionary entries may be generated as required. Constant verification and updating, if done manually, requires a lot of time which may be saved through automation.

DeMarco (in sec. 14.1.1 of [DEM78]) outlines eight features he feels are required of an 'ideal Data Dictionary processor' (see Fig. 8.1). PSL/PSA complies with most (if not all) of the requirements listed.

Other advantages appear at the Data Flow Diagram stage of Structured Analysis. Roughly sketched DFD's may be described using PSL and examined for consistency and completeness using reports such as the Data Activity Interaction Report and the Picture Report. This allows the system description to be built up as the analysis phase proceeds. Again, having the process automated rather than manual is to the analyst's advantage.

The purpose of this application was to examine the possibility of using PSL/PSA as an aid for Structured Analysis. Through observing its ease of adaptability and the advantages of automation this application of PSL/PSA is seen to be not only possible but also desirable.

There are a number of directions which could be pursued as the result of this study. The list includes:

1. A more comprehensive examination of PSA reports in terms of their usefulness.

2. The application of PSL/PSA and Structured Analysis to a live example rather than a case study.

3. The possibility of improving the DFD drawing capability of PSL/PSA (eg. Network (NETW) Report of V5.2).

4. The writing of specific reports (eg. to do logical generalization) for the use of Structured Analysis.

COMPARISON OF PSL/PSA TO DEMARCO'S IDEAL

REQUIREMENTS FOR AN IDEAL DATA DICTIONARY PROCESSOR	ABILITIES OF PSL/PSA
1. Accept definitions as input. Support the four classes of items - data flows, data elements, files and processes - that we have identified as essential.	PSL may be used to define all four classes of items as input to the stored system description.
2. Provide definition formats and procedures that adhere reasonably closely to the conventions of DeMarco's data dictionary.	Four PSA reports are able to mimic the data dictionary format required: Contents-report, Process-Summary-report, Dictionary-report, and Name-selection (which provides a complete directoy, or index, of all names).
3. Allow totally non-redundant input (if you specify that X is a subordinate of A, you should not be obliged to specify that A is a superordinate of X).	PSA allows totally non-redundant input (If A DERIVES B, then B DERIVED BY A need not be specified).
4. Allow easy updating of definitions.	The PSA data base is easily updated.
5. Supply some rudimentary consistency checking (included in this category are duplicate uses of names, disconnected aliases, circular definitions, syntactically incorrect definitions, and so forth).	PSA reports and syntax-checking abilities uncover duplicate names (e.g. Name-selection-report), disconnected aliases, and circular definitions.
6. Produce, as output, definition listings in alphabetic order by item name (For ease of maintenance, definitions should be presented in the same format as was used in the original input).	This is fulfilled by PSA reports (Contents-report, Name-selection-report with order=Alpha).
7. Provide facilities for alias control.	All PSL names and synonyms are unique.
8. Provide some elementary cross reference listings (For example, correlations of data elements to their superordinate data flows, listings of data elements required by each process, and listings of undefined terms.	Listings of data elements and undefined terms are available with many PSA reports. Sophisticated cross-referencing, completeness, and consistency checking is one of the principle strengths of PSL/PSA.

FIGURE 8.1

8.3 PSL/PSA

As a result of these applications of PSL/PSA, it is clear that PSL/PSA is more than merely an automated documentation tool. It is also able to provide a "systems view" model, not only of a system, but also of a model of a system as derived by some other techniques, in this case BIAIT and Structured Analysis. The application of PSL/PSA is not difficult once the model of the system is seen according to its objects, properties and relationships. The reporting capabilities of PSL/PSA are useful for maintaining up-to-date models, as well as for their diagnostic capabilities and the ability to observe the effects of changes resulting from altered model descriptions.

APPENDIX I

SET OF THREE BIAIT BICMX'S

Model PRODUCT DEFINITION 800	Management						Development			Engineer.												Service			Distribu.			Facilities			Personnel			Administr.	
	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL
Forecast market demand for product function 801	101 1101	803	805 805				807 804 803		808 808										303				207												
Specify particular product characteristics and quantities 802	101	102 805	803 805			811	805 804			709		902 902							801 801																
Select a development site 803					302		802 807 806												801									501		500					
Schedule development 804		803	104 106			811	802 802												801																
Review development activities 805		107			307		802 802	806	808 808			902 902	901 901						801 1002 801 1002																
Add accepted specs to the inventory of specifications 806			803			811																													
Evaluate specifications usage and performance 807		107 1003 1003			307	811			808			902 908											207												
Upgrade product specification as necessary 808			108 109 805 805			811		807		701		902							801 801																
809																																			
Transfer product specs to manufacturing 810		107 110	106			811						902 902																							
Account for development costs 811					307		802 804	806 807	808 810																										

103

Generic Model	General Management			Finance			Product Development			Manufac. Engineer.			Manufactur.			Purchasing			Marketing			Sales/ Service			Warehouse/ Distribu.			Facilities			Personnel			Administrat		
WORK IN PROCESS 900	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS
Establish production requirements 901		1003 1003	805 805							701	702		601	907				604 704	1001 1001			1002														
Develop work specifications and routings 902			805 805			911	802 802	807	808 810 810			908		907 903 906 907 910 1001																						
Pull the bill of material 903												902		904, 905 906 909 910 911						1004																
Order materials 904												902		607 903 610 903	911			604 704												710						
Validate completeness of materials for work to be done 905														607, 903, 906	610, 904, 906																					
Assign and initiate work 906														610, 905 903	903, 1000												1006 1010									
Evaluate the production process 907		107										902 902 908	901	906 909 911										1107						709						
Change work specifications and routings to improve the process 908								807				902		902																						
Re-route work in process due to problems 909														903 907 909 911																710						
Transfer materials and products 910												902		903 909 911													1006 1010									
Account for materials, time, tools and costs		111										902		904 907 910																						

Model	General Management			Finance			Product Development			Manfac. Engineer.			Manfactur.			Purchasing			Marketing			Sales/ Service			Warehouse/ Distribu.			Facilities			Personnel			Administ	
FINISHED GOODS 1000	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL	OPNS	PLNG	CNTL
Forecast demand for "finished goods" 1001	101 →	102 →					802 →						901 → 901						↑			1002	1107 ←					501							
Select the most appropriate finished goods to market 1002		1003	805 → 805				802 →						901						1001 →	207 → 204		↑	1103												
Select the source for finished goods 1003		↑			302 ← 307			807 ← 807		701 ← 701			601 901 ← 601 901			603 ← 703								1109		1007									
Order finished goods 1004						1011								903						↑				1106 ←		1007 ←									
Inspect received goods 1005						1011						902 →		↑													1006								
Warehouse finished goods 1006														906 910 1005 1008 1009 →												1007	↑								
Evaluate finished goods inventory usage 1007		1003 → 107				1011								1008 1009						1004			1109	1104		↑	1006 1010 ←								
Upgrade finished goods in inventory 1008						1011						902 →		906 ↑												1007 ←	1006								
Repair finished goods in inventory 1009						1011								↑	1009											1007 ←	1006								
Provide finished goods to the customer 1010						1011								906 910 →										1110		1007	↑								
Account for finished goods 1011		111			307									1003 1008 1009 ←						1004 ←						1007 ←	1010 ←								

APPENDIX II

DETAILS OF PSL/PSA USAGE

Once the above BIAIT -> PLS/PSA translation has been worked out for a specific application, all that remains to do is enter this information into the PSL/PSA database for this application (to be called BIAIT.DB for our example). This is a rather mechanical data entry procedure. One can enter the data directly into PSL/PSA's input program (called PSL); however, we find that the use of a text editor to enter the initial input useful, as PSL does not have extensive data editing features. As we are using the IBM 3031, we use SPF, creating a text file which is then designated to PSL as the "input file" to be used.

The form of entry into the SPF file is exactly as shown on page 5. When the text file has been created the sequence of entry commands are as follows:

Command	Explanation
1. ISDINIT BIAIT.DB	Initializes the DB file (a new DB for each applic.)
2. ISDPSA	Sets up the PSL/PSA system
3. SET DB=BIAIT.DB	Communicating the DB name to PSL
4. IPSL I=PSL.INPUT	Checks input file (PSL.INPUT) for system errors
5. IPSL I=PSL.INPUT U	If no errors, update file into the PSL/PSA DB

Upon completing these steps, the definitions contained in the text file will have been checked and entered into the named data base (BIAIT.DB). A variety of reports (via PSA) can now be generated, including the network diagram.

It should be noted that for any subsequent sessions having to do with the same application DB, only commands 2 and 3 need to be executed in order to generate reports, unless the information in the DB is to be updated.

APPENDIX III

PSA - BIAIT REPORTS

```
        */

   /* UNDEFINED NAME -                           SP202-801
      EMPLOYED BY:    A801;
        */

   /* UNDEFINED NAME -                           SP207-1002
      EMPLOYED BY:    A1002;
        */

   /* UNDEFINED NAME -                           SP207-801
      EMPLOYED BY:    A801;
        */

   /* UNDEFINED NAME -                           SP207-807
      EMPLOYED BY:    A807;
        */

   /* UNDEFINED NAME -                           SP302-1003
      EMPLOYED BY:    A1003;
        */

   /* UNDEFINED NAME -                           SP302-803
      EMPLOYED BY:    A803;
        */

   /* UNDEFINED NAME -                           SP307-805
      EMPLOYED BY:    A805;
        */

   /* UNDEFINED NAME -                           SP307-807
      EMPLOYED BY:    A807;
        */

   /* UNDEFINED NAME -                           SP601-1003
      EMPLOYED BY:    A1003;
        */

   /* UNDEFINED NAME -                           SP603-1003
      EMPLOYED BY:    A1003;
        */

   /* UNDEFINED NAME -                           SP607-904
      EMPLOYED BY:    A904;
        */

   /* UNDEFINED NAME -                           SP607-905
      EMPLOYED BY:    A905;
        */

   /* UNDEFINED NAME -                           SP610-905
      EMPLOYED BY:    A905;
        */

   /* UNDEFINED NAME -                           SP610-906
```

FORMATTED PROBLEM STATEMENT

```
      EMPLOYED BY:    A906;
        */
   /* UNDEFINED NAME -                   SP701-1003
      EMPLOYED BY:    A1003;
        */
   /* UNDEFINED NAME -                   SP707-901
      EMPLOYED BY:    A901;
        */
 DEFINE ENTITY                           E1001;
      SYNONYMS ARE:   FINISHED-GOODS-DEMAND;
      SUBPARTS ARE:   SP1001-501,
                      SP1001-901,
                      SP1001-1002;
      DERIVED BY:     A1001;
 DEFINE ENTITY                           E1002;
      SYNONYMS ARE:   FINISHED-GOOD-TO-MARKET-REPORT;
      SUBPARTS ARE:   SP1002-204,
                      SP1002-805,
                      SP1002-901,
                      SP1002-1003,
                      SP1002-1103;
      DERIVED BY:     A1002;
 DEFINE ENTITY                           E1003;
      SYNONYMS ARE:   FINISHED-GOODS-SOURCE-REPORT;
      SUBPARTS ARE:   SP1003-307,
                      SP1003-601,
                      SP1003-701,
                      SP1003-703,
                      SP1003-807,
                      SP1003-901,
                      SP1003-1007,
                      SP1003-1109;
      DERIVED BY:     A1003;
 DEFINE ENTITY                           E1004;
      SYNONYMS ARE:   FINISHED-GOODS-ORDER;
      SUBPARTS ARE:   SP1004-903,
                      SP1004-1011;
      DERIVED BY:     A1004;
 DEFINE ENTITY                           E1005;
      SYNONYMS ARE:   RECEIVED-GOODS-INSPECTION;
      SUBPARTS ARE:   SP1005-1006,
                      SP1005-1011;
      DERIVED BY:     A1005;
 DEFINE ENTITY                           E1006;
      SYNONYMS ARE:   WAREHOUSE-GOODS-REPORT;
      SUBPARTS ARE:   SP1006-1007;
      DERIVED BY:     A1006;
```

FORMATTED PROBLEM STATEMENT

```
DEFINE ENTITY                                   SP1003-703;
     PART OF:           E1003;
     DERIVED BY:        A1003;

DEFINE ENTITY                                   SP1003-807;
     PART OF:           E1003;
     DERIVED BY:        A1003;
     EMPLOYED BY:       A807;

DEFINE ENTITY                                   SP1003-901;
     PART OF:           E1003;
     DERIVED BY:        A1003;
     EMPLOYED BY:       A901;

DEFINE ENTITY                                   SP1004-1011;
     PART OF:           E1004;
     DERIVED BY:        A1004;
     EMPLOYED BY:       A1011;

DEFINE ENTITY                                   SP1004-903;
     PART OF:           E1004;
     DERIVED BY:        A1004;
     EMPLOYED BY:       A903;

DEFINE ENTITY                                   SP1005-1006;
     PART OF:           E1005;
     DERIVED BY:        A1005;
     EMPLOYED BY:       A1006;

DEFINE ENTITY                                   SP1005-1011;
     PART OF:           E1005;
     DERIVED BY:        A1005;
     EMPLOYED BY:       A1011;

DEFINE ENTITY                                   SP1006-1007;
     PART OF:           E1006;
     DERIVED BY:        A1006;
     EMPLOYED BY:       A1007;

DEFINE ENTITY                                   SP1007-1004;
     PART OF:           E1007;
     DERIVED BY:        A1007;
     EMPLOYED BY:       A1004;

DEFINE ENTITY                                   SP1007-1008;
     PART OF:           E1007;
     DERIVED BY:        A1007;
     EMPLOYED BY:       A1008;

DEFINE ENTITY                                   SP1007-1009;
     PART OF:           E1007;
     DERIVED BY:        A1007;
     EMPLOYED BY:       A1009;

DEFINE ENTITY                                   SP1007-1011;
```

```
                    SP902-807,
                    SP908-807,
                    SP1003-807;

DEFINE PROCESS                          A808;
    SYNONYMS ARE:   UPGRADE-PRODUCT-SPECS;
    DESCRIPTION;
     THIS PROCESS UPGRADES THE PRODUCT SPECIFICATIONS AS
     NECESSARY.;
    DERIVES:        E808;
    DERIVES:        SP808-701;
    DERIVES:        SP808-801;
    DERIVES:        SP808-805;
    DERIVES:        SP808-811;
    DERIVES:        SP808-902;
    EMPLOYS:        SP108-808,
                    SP109-808,
                    SP801-808,
                    SP805-808,
                    SP807-808;

DEFINE PROCESS                          A810;
    SYNONYMS ARE:   TRANSFER-PROD-SPECS;
    DESCRIPTION;
     THIS PROCESS TRANSFERS THE PRODUCT SPECIFICATIONS TO
     MANUFACTURING.;
    DERIVES:        E810;
    DERIVES:        SP810-106;
    DERIVES:        SP810-107;
    DERIVES:        SP810-110;
    DERIVES:        SP810-811;
    DERIVES:        SP810-902;
    EMPLOYS:        SP902-810;

DEFINE PROCESS                          A811;
    SYNONYMS ARE:   ACCOUNT-FOR-DEV-COSTS;
    DESCRIPTION;
     THIS PROCESS ACCOUNTS FOR DEVELOPEMENT COSTS.;
    DERIVES:        E811;
    DERIVES:        SP811-307;
    EMPLOYS:        SP802-811,
                    SP804-811,
                    SP806-811,
                    SP807-811,
                    SP808-811,
                    SP810-811;

DEFINE PROCESS                          A901;
    SYNONYMS ARE:   ESTABLISH-PROD-REQUIREMENTS;
    DESCRIPTION;
     THIS PROCESS ESTABLISHES PRODUCTION REQUIREMENT6.;
    DERIVES:        E901;
    DERIVES:        SP901-601;
    DERIVES:        SP901-604;
    DERIVES:        SP901-701;
```

FORMATTED PROBLEM STATEMENT

```
     DERIVES:        SP901-704;
     DERIVES:        SP901-805;
     DERIVES:        SP901-1001;
     DERIVES:        SP901-1003;
     EMPLOYS:        SP707-901,
                     SP805-901,
                     SP907-901,
                     SP1001-901,
                     SP1002-901,
                     SP1003-901;

DEFINE PROCESS                           A902;
     SYNONYMS ARE:   DEVELOP-WORK-SPEC-AND-ROUTINGS;
     DESCRIPTION;
      THIS PROCESS DEVELOPES WORK SPECIFICATIONS AND ROUTINGS.;
     DERIVES:        E902;
     DERIVES:        SP902-802;
     DERIVES:        SP902-805;
     DERIVES:        SP902-807;
     DERIVES:        SP902-810;
     DERIVES:        SP902-903;
     DERIVES:        SP902-904;
     DERIVES:        SP902-907;
     DERIVES:        SP902-910;
     DERIVES:        SP902-1008;
     DERIVES:        SP902-1005;
     EMPLOYS:        SP802-902,
                     SP805-902,
                     SP808-902,
                     SP810-902,
                     SP907-902,
                     SP908-902,
                     SP911-902;

DEFINE PROCESS                           A903;
     SYNONYMS ARE:   PULL-THE-BILL-OF-MATERIAL;
     DESCRIPTION;
      THIS PROCESS PULLS THE BILL OF MATERIAL.;
     DERIVES:        E903;
     DERIVES:        SP903-904;
     DERIVES:        SP903-905;
     DERIVES:        SP903-906;
     DERIVES:        SP903-909;
     DERIVES:        SP903-910;
     DERIVES:        SP903-911;
     EMPLOYS:        SP902-903,
                     SP1004-903;

DEFINE PROCESS                           A904;
     SYNONYMS ARE:   ORDER-MATERIALS;
     DESCRIPTION;
      THIS PROCESS ORDERS MATERIALS.;
     DERIVES:        E904;
     DERIVES:        SP904-604;
     DERIVES:        SP904-610;
```

DATA ACTIVITY INTERACTION REPORT

THE ROWS ARE DATA NAMES, THE COLUMNS ARE ACTIVITY NAMES.

ROW NAMES

```
 1 SP101-1001        *** UNDEFINED ***
 2 E1001             ENTITY
 3 SP1001-501        ENTITY
 4 SP1001-901        ENTITY
 5 SP1001-1002       ENTITY
 6 SP102-1001        *** UNDEFINED ***
 7 SP802-1001        ENTITY
 8 SP901-1001        ENTITY
 9 SP1107-1001       *** UNDEFINED ***
10 E901              ENTITY
11 SP901-601         ENTITY
12 SP901-604         ENTITY
13 SP901-701         ENTITY
14 SP901-704         ENTITY
15 SP901-805         ENTITY
16 SP901-1003        ENTITY
17 SP707-901         *** UNDEFINED ***
18 SP805-901         ENTITY
19 SP907-901         ENTITY
20 SP1002-901        ENTITY
21 SP1003-901        ENTITY
22 E1002             ENTITY
23 SP1002-204        ENTITY
24 SP1002-805        ENTITY
25 SP1002-1003       ENTITY
26 SP1002-1103       ENTITY
27 SP207-1002        *** UNDEFINED ***
28 SP802-1002        ENTITY
29 SP805-1002        ENTITY
30 E802              ENTITY
31 SP802-701         ENTITY
32 SP802-801         ENTITY
33 SP802-803         ENTITY
34 SP802-804         ENTITY
35 SP802-805         ENTITY
36 SP802-811         ENTITY
37 SP802-902         ENTITY
38 SP101-802         *** UNDEFINED ***
39 SP102-802         *** UNDEFINED ***
40 SP801-802         ENTITY
41 SP804-802         ENTITY
42 SP805-802         ENTITY
43 SP902-802         ENTITY
44 SP803-802         ENTITY
45 E805              ENTITY
46 SP805-801         ENTITY
47 SP805-806         ENTITY
48 SP805-808         ENTITY
49 SP805-902         ENTITY
50 SP107-805         *** UNDEFINED ***
```

DATA ACTIVITY INTERACTION REPORT

THE ROWS ARE DATA NAMES, THE COLUMNS ARE ACTIVITY NAMES.

COLUMN NAMES

1	A1001	PROCESS
2	A901	PROCESS
3	A1002	PROCESS
4	A802	PROCESS
5	A805	PROCESS
6	A1003	PROCESS
7	A907	PROCESS
8	A801	PROCESS
9	A803	PROCESS
10	A804	PROCESS
11	A811	PROCESS
12	A902	PROCESS
13	A806	PROCESS
14	A808	PROCESS
15	A807	PROCESS
16	A1007	PROCESS
17	A908	PROCESS
18	A909	PROCESS
19	A911	PROCESS
20	A906	PROCESS
21	A810	PROCESS
22	A903	PROCESS
23	A904	PROCESS
24	A910	PROCESS
25	A1008	PROCESS
26	A1005	PROCESS
27	A1004	PROCESS
28	A1009	PROCESS
29	A1011	PROCESS
30	A1006	PROCESS
31	A1010	PROCESS
32	A905	PROCESS

DATA ACTIVITY INTERACTION REPORT

DATA ACTIVITY INTERACTION MATRIX

(I,J) VALUE	MEANING
R	ROW I IS RECEIVED OR USED BY COLUMN J (INPUT)
U	ROW I IS UPDATED BY COLUMN J
D	ROW I IS DERIVED OR GENERATED BY COLUMN J (OUTPUT)
A	ROW I IS INPUT TO, UPDATED BY, AND OUTPUT OF COLUMN J (ALL)
F	ROW I IS INPUT TO AND OUTPUT OF COLUMN J (FLOW)
1	ROW I IS INPUT TO AND UPDATED BY COLUMN J
2	ROW I IS UPDATED BY AND OUTPUT OF COLUMN J

```
             11111111112222222222333
    12345678901234567890123456789012
   +----+----+----+----+----+----+--+
 1 |R   |    |    |    |    |    |  |
 2 |D   |    |    |    |    |    |  |
 3 |D   |    |    |    |    |    |  |
 4 |DR  |    |    |    |    |    |  |
 5 +D-R-+----+----+----+----+----+--+
 6 |R   |    |    |    |    |    |  |
 7 |R  D|    |    |    |    |    |  |
 8 |RD  |    |    |    |    |    |  |
 9 |R   |    |    |    |    |    |  |
10 +-D--+----+----+----+----+----+--+
11 | D  |    |    |    |    |    |  |
12 | D  |    |    |    |    |    |  |
13 | D  |    |    |    |    |    |  |
14 | D  |    |    |    |    |    |  |
15 +-D--R----+----+----+----+----+--+
16 | D  |R   |    |    |    |    |  |
17 | R  |    |    |    |    |    |  |
18 | R  D    |    |    |    |    |  |
19 | R  | D  |    |    |    |    |  |
20 +-RD-+----+----+----+----+----+--+
21 | R  |D   |    |    |    |    |  |
22 |  D |    |    |    |    |    |  |
23 |  D |    |    |    |    |    |  |
24 |  D R    |    |    |    |    |  |
25 |  D |R   |    |    |    |    |  |
   +----+----+----+----+----+----+--+
```

116

MC MASTER - ISRAM PSA PROJECT

DATA ACTIVITY INTERACTION REPORT

DATA ACTIVITY INTERACTION MATRIX

```
             11111111112222222222333
    12345678901234567890123456789012
   +----+----+----+----+----+----+--+
26 |  D |    |    |    |    |    |  |
27 |  R |    |    |    |    |    |  |
28 |  RD|    |    |    |    |    |  |
29 |  R D    |    |    |    |    |  |
30 +---D+----+----+----+----+----+--+
31 |   D|    |    |    |    |    |  |
32 |   D|  R |    |    |    |    |  |
33 |   D|   R|    |    |    |    |  |
34 |   D|    R    |    |    |    |  |
35 +---DR----+----+----+----+----+--+
36 |   D|    |R   |    |    |    |  |
37 |   D|    | R  |    |    |    |  |
38 |   R|    |    |    |    |    |  |
39 |   R|    |    |    |    |    |  |
40 +---R+--D-+----+----+----+----+--+
41 |   R|    D    |    |    |    |  |
42 |   RD    |    |    |    |    |  |
43 |   R|    | D  |    |    |    |  |
44 |   R|   D|    |    |    |    |  |
45 +----D----+----+----+----+----+--+
46 |    D  R |    |    |    |    |  |
47 |    D    |  R |    |    |    |  |
48 |    D    |   R|    |    |    |  |
49 |    D    | R  |    |    |    |  |
50 |    R    |    |    |    |    |  |
   +----+----+----+----+----+----+--+
```

-17

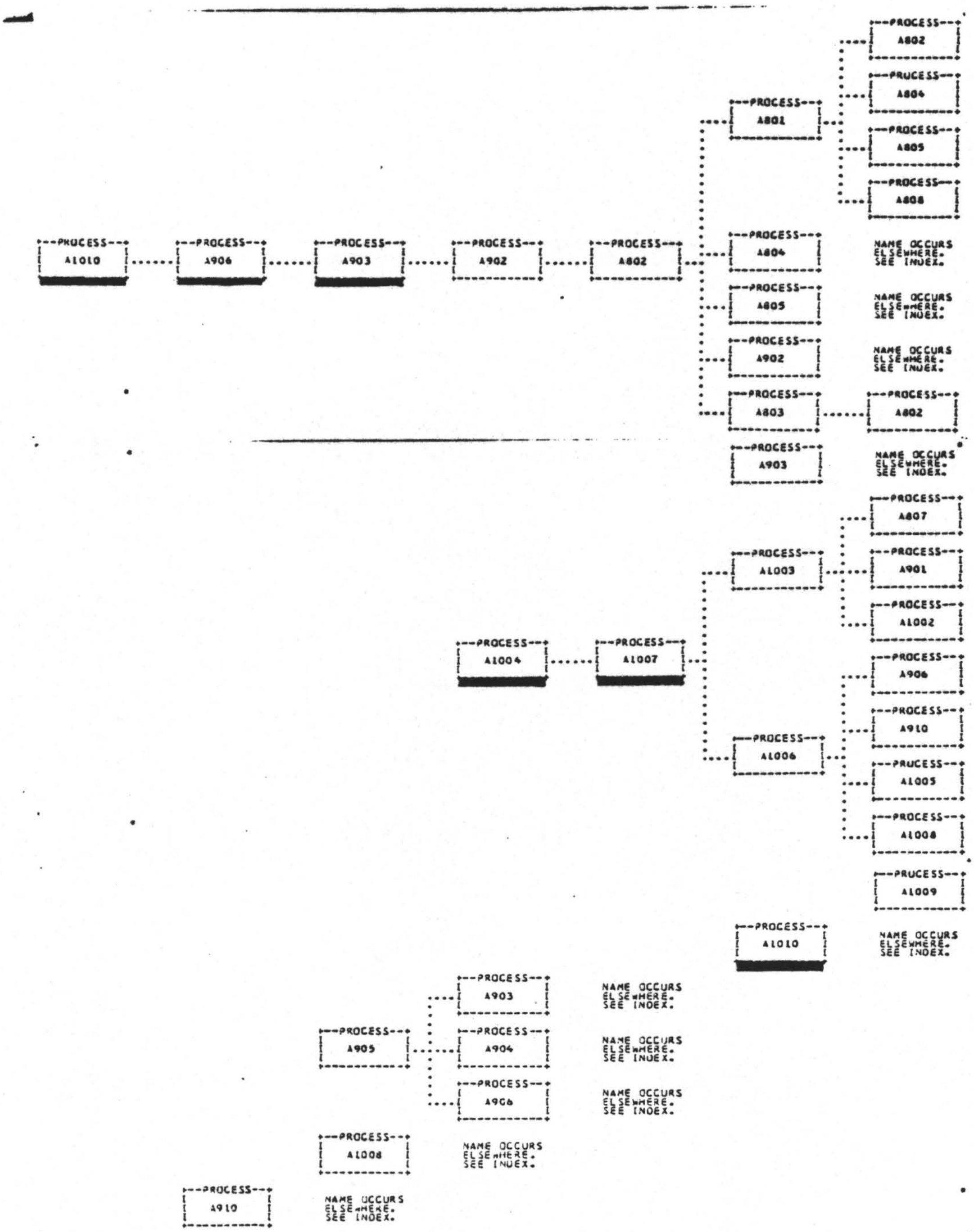
DATA ACTIVITY INTERACTION REPORT

DATA ACTIVITY INTERACTION MATRIX ANALYSIS

DATA			
E1001	(ENTITY)	(ROW 2)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1001-501	(ENTITY)	(ROW 3)	DERIVED, BUT NOT USED BY ANY PROCESS
E901	(ENTITY)	(ROW 10)	DERIVED, BUT NOT USED BY ANY PROCESS
SP901-601	(ENTITY)	(ROW 11)	DERIVED, BUT NOT USED BY ANY PROCESS
SP901-604	(ENTITY)	(ROW 12)	DERIVED, BUT NOT USED BY ANY PROCESS
SP901-701	(ENTITY)	(ROW 13)	DERIVED, BUT NOT USED BY ANY PROCESS
SP901-704	(ENTITY)	(ROW 14)	DERIVED, BUT NOT USED BY ANY PROCESS
E1002	(ENTITY)	(ROW 22)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1002-204	(ENTITY)	(ROW 23)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1002-1103	(ENTITY)	(ROW 26)	DERIVED, BUT NOT USED BY ANY PROCESS
E802	(ENTITY)	(ROW 30)	DERIVED, BUT NOT USED BY ANY PROCESS
SP802-701	(ENTITY)	(ROW 31)	DERIVED, BUT NOT USED BY ANY PROCESS
E805	(ENTITY)	(ROW 45)	DERIVED, BUT NOT USED BY ANY PROCESS
E1003	(ENTITY)	(ROW 55)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1003-307	(ENTITY)	(ROW 56)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1003-601	(ENTITY)	(ROW 57)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1003-701	(ENTITY)	(ROW 58)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1003-703	(ENTITY)	(ROW 59)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1003-1109	(ENTITY)	(ROW 62)	DERIVED, BUT NOT USED BY ANY PROCESS
E907	(ENTITY)	(ROW 68)	DERIVED, BUT NOT USED BY ANY PROCESS
SP907-107	(ENTITY)	(ROW 69)	DERIVED, BUT NOT USED BY ANY PROCESS
SP907-709	(ENTITY)	(ROW 70)	DERIVED, BUT NOT USED BY ANY PROCESS
SP907-1107	(ENTITY)	(ROW 75)	DERIVED, BUT NOT USED BY ANY PROCESS
E801	(ENTITY)	(ROW 78)	DERIVED, BUT NOT USED BY ANY PROCESS
E803	(ENTITY)	(ROW 88)	DERIVED, BUT NOT USED BY ANY PROCESS
SP803-501	(ENTITY)	(ROW 89)	DERIVED, BUT NOT USED BY ANY PROCESS
SP803-508	(ENTITY)	(ROW 90)	DERIVED, BUT NOT USED BY ANY PROCESS
E804	(ENTITY)	(ROW 93)	DERIVED, BUT NOT USED BY ANY PROCESS
SP804-106	(ENTITY)	(ROW 94)	DERIVED, BUT NOT USED BY ANY PROCESS
E811	(ENTITY)	(ROW 97)	DERIVED, BUT NOT USED BY ANY PROCESS
SP811-307	(ENTITY)	(ROW 98)	DERIVED, BUT NOT USED BY ANY PROCESS
E902	(ENTITY)	(ROW 103)	DERIVED, BUT NOT USED BY ANY PROCESS
E806	(ENTITY)	(ROW 115)	DERIVED, BUT NOT USED BY ANY PROCESS
E808	(ENTITY)	(ROW 116)	DERIVED, BUT NOT USED BY ANY PROCESS
SP808-701	(ENTITY)	(ROW 117)	DERIVED, BUT NOT USED BY ANY PROCESS
E807	(ENTITY)	(ROW 121)	DERIVED, BUT NOT USED BY ANY PROCESS
E1007	(ENTITY)	(ROW 126)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1007-107	(ENTITY)	(ROW 127)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1007-1104	(ENTITY)	(ROW 132)	DERIVED, BUT NOT USED BY ANY PROCESS
SP1007-1107	(ENTITY)	(ROW 133)	DERIVED, BUT NOT USED BY ANY PROCESS
E908	(ENTITY)	(ROW 136)	DERIVED, BUT NOT USED BY ANY PROCESS
E909	(ENTITY)	(ROW 137)	DERIVED, BUT NOT USED BY ANY PROCESS
SP909-710	(ENTITY)	(ROW 138)	DERIVED, BUT NOT USED BY ANY PROCESS
E911	(ENTITY)	(ROW 141)	DERIVED, BUT NOT USED BY ANY PROCESS
SP911-111	(ENTITY)	(ROW 142)	DERIVED, BUT NOT USED BY ANY PROCESS
E906	(ENTITY)	(ROW 146)	DERIVED, BUT NOT USED BY ANY PROCESS
E810	(ENTITY)	(ROW 154)	DERIVED, BUT NOT USED BY ANY PROCESS
SP810-106	(ENTITY)	(ROW 155)	DERIVED, BUT NOT USED BY ANY PROCESS
SP810-107	(ENTITY)	(ROW 156)	DERIVED, BUT NOT USED BY ANY PROCESS
SP810-110	(ENTITY)	(ROW 157)	DERIVED, BUT NOT USED BY ANY PROCESS

APPENDIX IV

EP - PRECEDENCE HANDLING REPORT


PROCESS A1010
PROCESS A906
PROCESS A903
PROCESS A902
PROCESS A802
PROCESS A801
PROCESS A802
PROCESS A804
PROCESS A805
PROCESS A808
PROCESS A804
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A805
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A902
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A803
PROCESS A802
PROCESS A903
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A1004
PROCESS A1007
PROCESS A1003
PROCESS A807
PROCESS A901
PROCESS A1002
PROCESS A1006
PROCESS A906
PROCESS A910
PROCESS A1005
PROCESS A1008
PROCESS A1009
PROCESS A1010
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A905
PROCESS A903
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A904
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A906
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A1008
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS A910
NAME OCCURS ELSEWHERE. SEE INDEX.

APPENDIX V

FROMAGE DU MIDI - CASE STUDY

CASE STUDY OVERVIEW

Background

The French Ministry of Commerce has established a system of national merchandise markets in each of the large metropolitan areas. These markets exist to facilitate trade between wholesalers and retailers in a large range of soft commodities (meats, vegetables, flowers, fabrics, wines, charcuterie, dairy products, etc.).

One of these markets is situated in the small village of Fleury les Deux Eglises, just outside of Lyons. There has been a market in the village since pre-Roman times. The current market was renovated in 1966. At that time, a modest computing system was added to assist in billing and accounting.

Market Operations

The market opens for business at midnight. At that time, only the wholesalers are present. They work until 5 am stocking the market. During this period, goods arrive at the truck docks around the periphery of the market and are accepted there by receiving clerks. Each arrival is documented and placed onto one or more conveyor system hangers. These hangers are then shunted onto the conveyor network and addressed to the termination point associated with the wholesaler who owns the shipment.

Buyers (retailers) begin to arrive at about 3 am. They first check in at the Caisse Centrale and establish credit for the day's purchases. This transaction may involve paying a past due bill or making a deposit or simple negotiation with the market manager. It is customary at this time, as well, to shake hands with everyone and then go have a bowl of onion soup and two or three glasses of red wine.

Suitably fortified, the retailers now begin their rounds of the market. A typical retailer will visit 20-30 different wholesale booths to inspect merchandise and haggle over prices. He may do business with as many of 15 wholesalers in a given day. Each time an agreement is made, there is an exchange of tickets to document the sale. The goods themselves are marked and set aside. All goods remain on the conveyor system hanger. The number of the hanger is placed on all the tickets. Ticket copies for each sale are sent to the Caisse Centrale by pneumatic tube.

When his purchases are complete, the retailer checks out at the Caisse Centrale. A composite statement for the entire day's purchases will be generated for him by noon of the same day. As part of the checkout process, a list of his purchases is sent to the dock manager. The retailer stops by the manager's office and receives instructions for pickup of his combined purchases. Instructions are in the form of a dock number and arrival time. He then goes to the cafe for another glass of red wine, or (if the negotiations have been particularly successful) a Pernod. The dock manager now sends conveyor leading instructions to each of the wholesale booths. He tells them which orders to shunt onto the system, when to start them off, and the dock address to send them to. At his appointed time, the retailer drives up to the allocated truck dock, just as his purchases are arriving by conveyor from the different wholesale booths.

The Conveyor System

The conveyor system at Fleury dates from the late nineteenth century. It is made totally of iron and weighs more than twice as much as the Eiffel tower. (It is affectionately known as "Le Gros Monstrueux" by the merchants.) The system consists of some 200 trucks and more than 2000 junction points. At each junction point, a particular hanger can select one of two possible directions. There are 900 terminations of the network. Each termination is associated with either a wholesaler location or a truck dock.

In spite of its age, the conveyor system has a complete stored address routing system. When a hanger is shunted onto the system, its terminating address is entered by placing a coded set of metal beads into a matrix of rubber tracks on the hanger. At each junction mechanical fingers sense the position of the beads and decide which direction to route the hanger. Less than one hanger in 5000 fails to arrive at its proper destination. Misaddressed or misdirected hangers are placed back on the network by the actual receiver.

Reason for the Current Study

Due to increased volume and the escalating cost of associated paperwork, the decision has been made to consider automating more of the market's operations. Since there is little or no documentation of existing procedures, the first part of the new project will be devoted to a study of current operations. The market management has agreed to approach analysis and new system specification in a structured fashion.

- -

DFD EXERCISE

As part of the study of current operations at the Fleury market, interviews were conducted with several wholesalers. The firm, Fromages du Midi is typical of these. It is a medium sized cheese broker, specializing in fermented cheeses such as bries and camemberts from the south of France. Reproduced below are notes from discussions with M. Le Grandcharles (the manager), and his assistants, M. Asterix and M. Obelix.

Interview with M. Obelix

Obelix is the first to come in each morning. He must be present by a few minutes past midnight when the first delivered lots arrive from the truck docks by conveyor. Each conveyor hanger arrives with cheese cases attached and a copy of the trucker's delivery document. When where there are several lots (several hangers) that make up a shipment, each one has its own documentation. Obelix pushes and shoves the loaded hanger into the sales area. (Pushing and shoving loaded hangers is Obelix's great talent.) Some lots are damaged or incomplete or incorrectly addressed. These Obelix shunts back onto the conveyor and addresses them back to the truck dock or to another wholesaler. If any lots are accepted, Obelix makes up a release and sends ti to the trucker via pneumatic tube.

When the buyers select merchandise, some regrouping of cases among the hangers may be required. Obelix also does this. Under the direction of M. Asterix, he shifts cases from one hanger to another.

At the time of sale, all ticket copies are removed from the hangers. From time to time, Obelix is asked to reticket sold hangers. He receives bundles of red tickets from M. Le G., and places one on each hanger (the red tickets are keyed to hanger number).

Whenever a pneumatic tube arrives, Obelix drops whatever he is doing and goes over to get the message. If it is a retailer's receipt, he gives it to M. Asterix. Otherwise it is a sending list from the Dock Manager. Obelix rushes the sending list over to M. Le G., since there may be hangers that have to be sent immediately. He waits for any immediate delivery tickets that M. Le G. may have for him. Each delivery ticket contains hanger number, time of departure, and truck dock destination. For each delivery ticket that is due for departure, Obelix does all the following:

- o find the proper hanger and affix delivery ticket to it
- o remove the red ticket and mark it "sent"
- o code the destination dock into the address matrix with metal beads
- o shunt the hanger onto the conveyor
- o give the sent ticket to M. Asterix.

The delivery tickets that are not yet due are placed on a large display board by M. Le Grandcharles. They are in order by time of day. Every five minutes, Obelix checks the board and sends out any hangers that have come due.

<u>Interview with M. Asterix</u>

Asterix does all the selling and handles some of the paper work which might normally fall to Obelix (who is not so good with paper). He checks through any delivered lots in the sales area looking for those that still have delivery documents attached. Each time he finds one, he marks the hanger number on it and places it in the Receipts File. He makes up a fresh set of tickets and places it on the hanger. One copy of each ticket he keeps for himself for his daily reports to the owner.

When customers come in, he guides them through the Sales area and negotiates with them. When they select goods, Asterix takes tickets from the hanger and marks conditions of sale on the top copy. The tickets are given to M. Le G. If no sale is made, Asterix takes the customer's copy of the order and marks it no sale and gives it back to M. Le G. If a customer wants part of a hanger, Asterix calls in Obelix to do the actual heavy work, while he limits himself to revising the tickets and updating the Receipts File so it will again show correct hanger numbers.

Before leaving, Asterix makes up three reports for the owner: Daily Sales Report (compiled from the sent goods tickets), Inventory Status Summary (compiled from the set of arrival tickets, the sent goods tickets and the retailer receipts), and a Delivery Time Report (compiled from the sent goods tickets and the time-stamped retailer receipts).

Interview with M. Le Grandcharles

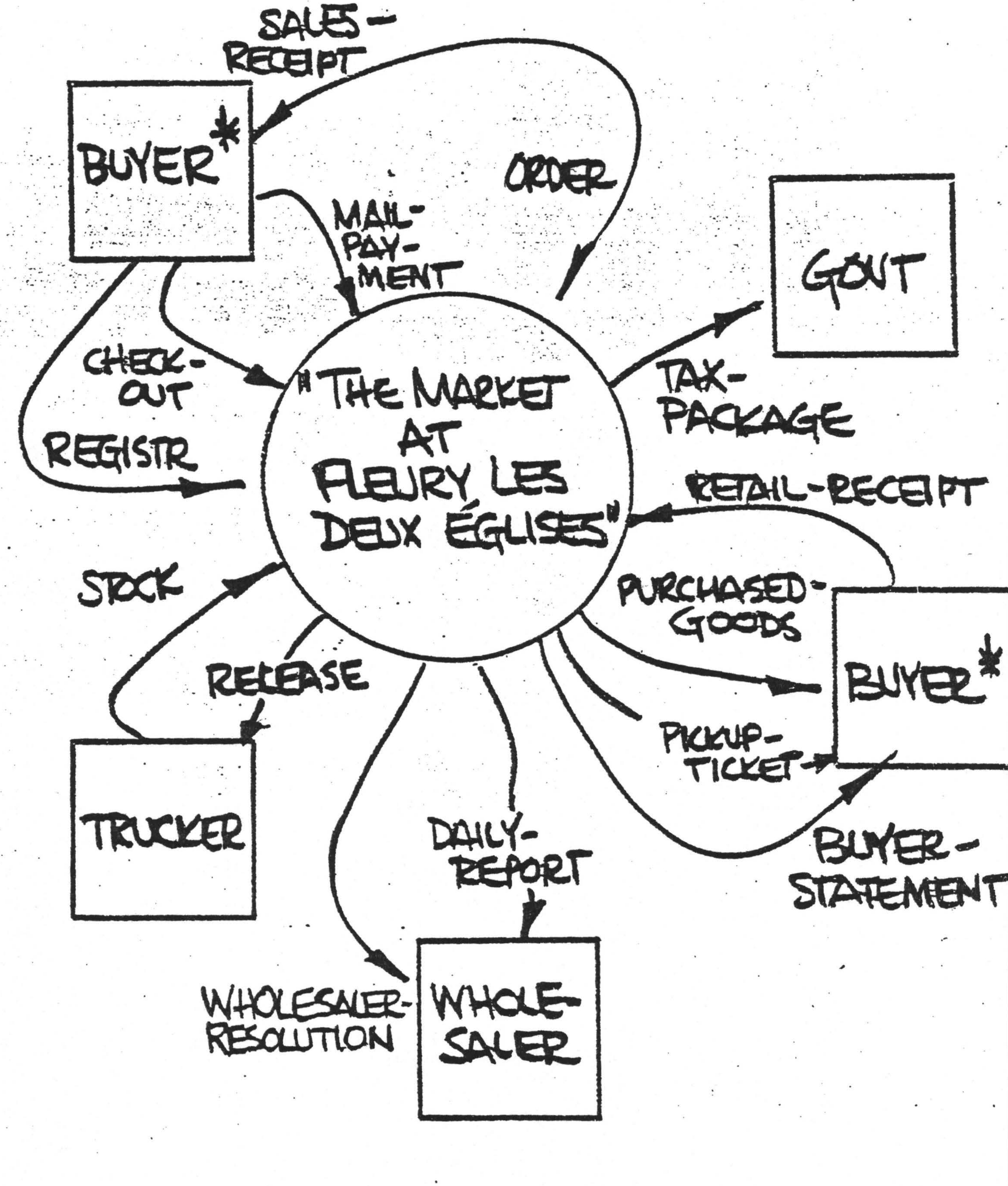
M. Le Grandcharles, being manager of the whole operation, makes a great point of never leaving his desk. He greets buyers as they come in and makes up a tentative order form for each one (although it is not certain that the buyer will actually buy anything yet). Based on the requirements stated in the tentative order, M. Le G. calculates an estimated total and calls down to the Caisse Centrale to hold that much of the buyer's purchasing power. If the confirmation comes back positive, the tentative order is initialed and sent on with the buyer into the sales area.

If the buyer makes no purchase, the order form comes back via M. Asterix, and that means it is necessary to call the Caisse again to release the buyer's funds. When a sale is made, M. Le G. revises the order, copying the actual quantities and amounts from the picked stock tickets onto the order. He gives a sales receipt and each of the yellow ticket copies to the buyer. The revised order is bundled together with the remaining tickets and set aside for later documentation.

When time permits, M. Le G. picks up the order packet (marked up order form and tickets) and uses it to type up a clean completed order. This he sends to the Caisse by pneumatic tube along with the green ticket copies. There are now two tickets left (a white and a red) associated with each sold hanger. The red ones he gives to Obelix to mark the hangers as sold. The white ones are attached to the order copy and filed.

When Obelix brings in a sending list, M. Le Grandcharles fetches the tickets for each order and copies time and destination onto each one. These are now "deliverable tickets," and he places them on the Deliverables Board for Obelix.

EXERCISE: Draw a Data Flow Diagram for Wholesale Operations.


SALES-
RECEIPT
BUYER*
ORDER
MAIL-
PAY-
MENT
GOVT
CHECK-
OUT
REGISTR
"THE MARKET
AT
FLEURY LES
DEUX ÉGLISES"
TAX-
PACKAGE
RETAIL-RECEIPT
STOCK
PURCHASED-
GOODS
RELEASE
BUYER*
PICKUP-
TICKET
TRUCKER
DAILY-
REPORT
BUYER-
STATEMENT
WHOLESALER-
RESOLUTION
WHOLE-
SALER

B.

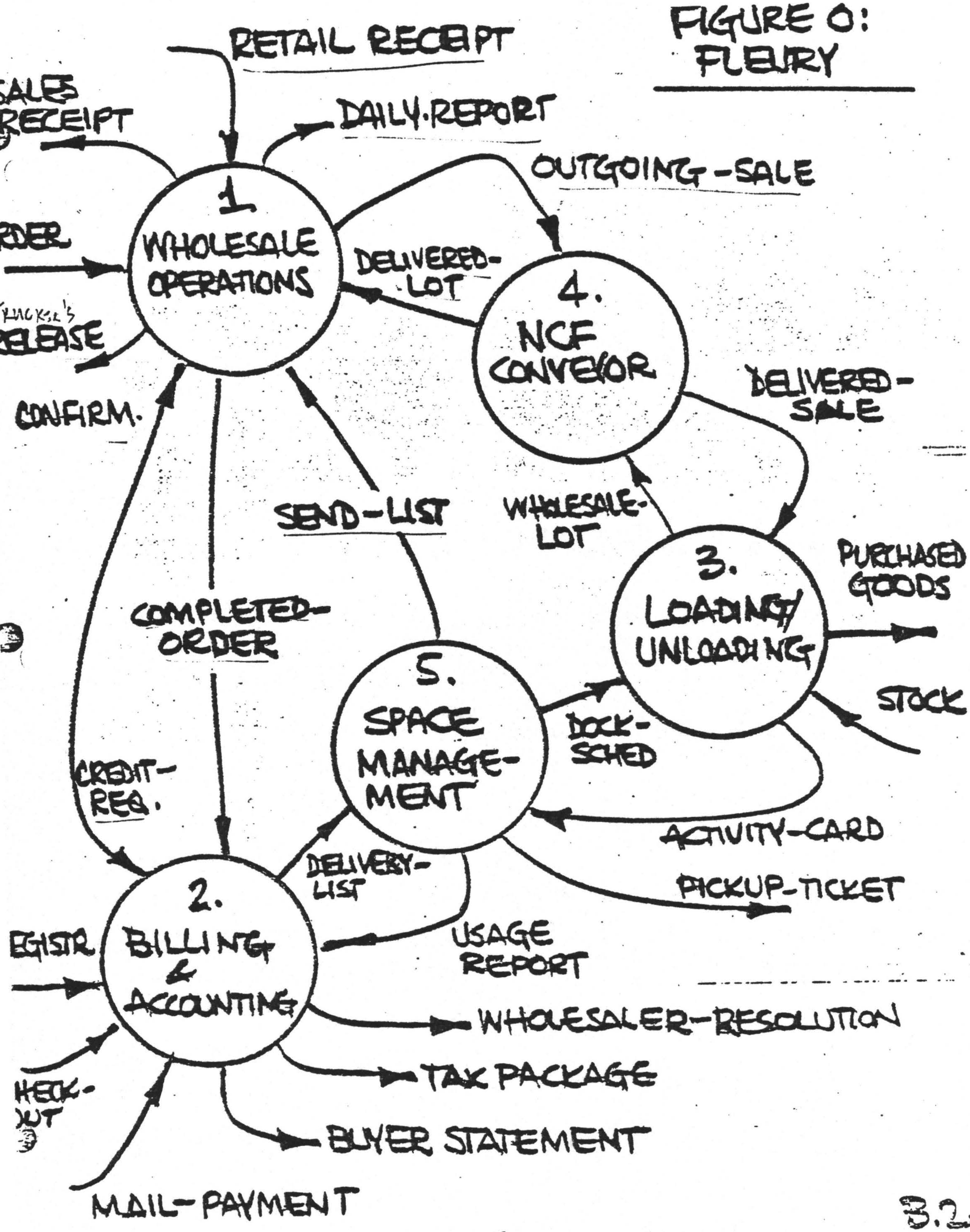


FIGURE O:
FLEURY
RETAIL RECEIPT
SALES RECEIPT
DAILY-REPORT
OUTGOING-SALE
1. WHOLESALE OPERATIONS
RDER
RUCKER'S
RELEASE
CONFIRM.
DELIVERED-LOT
4. NCF CONVEYOR
DELIVERED-SALE
SEND-LIST
WHOLESALE-LOT
3. LOADING/UNLOADING
PURCHASED GOODS
COMPLETED-ORDER
STOCK
5. SPACE MANAGEMENT
DOCK-SCHED
CREDIT-REQ.
ACTIVITY-CARD
DELIVERY-LIST
PICKUP-TICKET
2. BILLING & ACCOUNTING
EGISTR
USAGE REPORT
WHOLESALER-RESOLUTION
HECK-OUT
TAX PACKAGE
BUYER STATEMENT
MAIL-PAYMENT
3.2

CASE STUDY DATA DICTIONARY

Accepted-Lot	=	*arrival at wholesaler booth*
	=	Hanger + Delivery-Document
Active-Credit-File	=	*buyer purchasing power worksheets*
	=	{Credit-Ledger + {Credit-Revision}}
Activity-Card	=	*current status update for dock*
	=	Dock-Number + Time + Wholesaler-ID + [Arriving-Buyer-ID | Departing-Buyer-ID]
Arrival-Line	=	*report detail about accepted lots*
	=	Hanger-Number + 1{Qty + Item-Description}
Buyer-Compl.	=	*transaction to close out buyer for day*
Buyer-Reg.	=	*buyer activation transaction*
Buyer-Statement	=	*composite invoice for day's purchases*
Check-Out	=	*departing buyer's notice of completion*
	=	Buyer-Name + {Order-Form}
Completed-Order	=	*final documentation of sale*
	=	Order-Form + {Filled-In-Ticket}
Confirm	=	*response on buyer's available credit*
	=	Buyer-Name + Buyer-ID + [Positive Acknowledgement | Negative-Acknowledgement]
Correction	=	*transaction to modify master file*
Credit-Amount	=	*transaction to record new purchaisng power*
Credit-Instr.	=	*manager's authorization to increase credit*
	=	Buyer-ID + Amount
Credit-Ledger	=	*worksheet for credits and holds*
Credit-Req.	=	*request to hold or release funds*
	=	[Buyer-Name | Buyer-ID] + Wholesaler-ID + Amount-to-hold *may be negative*
Credit-Revision	=	*annotation to credit ledger*
	=	[Wholesaler-ID + Amount-Held | Buyer-ID + Amount-Paid | Buyer-ID + Amount-of-Added-Credit] + New-Credit-Amount + Initials
Credit-Trans	=	*modify retailer's purchasing power*
	=	[Hold | Payment | Credit-Amount]

Credited-Order-Form	=	*documentation of authorized credit*
	=	Order-Form + Held-Amount + Signature
Daily-Report	=	*summary of wholesale operations*
	=	Daily-Sales-Report + Inventory-Status-Summary + Delivery-Time-Report
Daily-Sales-Report	=	*wholesaler repared summary*
	=	Date + Sales-Total {Qty + Item-Description + Buyer-ID + Amount}
Delivered-Lot	=	*goods arriving at wholesaler booth*
	=	Hanger + Delivery-Document
Delivered-Sale	=	*sold goods arriving at docks*
	=	Delivery-Ticket + Hanger
Deliveries-File	=	*orders awaiting buyer completion*
	=	Buyer-ID + 1{Order-Number}
Delivery-Document	=	*trucker's description of shipment*
	=	Trucker-ID + Originating-Dealer-Name + 1{Qty + Item-Description}
Delivery-Line	=	*report detail on dispatched goods*
	=	Hanger-Number + Buyer-ID
Delivery-List	=	*list of orders to be dispatched*
	=	Buyer-ID + {Wholesaler-ID + {Order-Number}}
Delivery-Ticket	=	*documentation of dispatchable hanger*
	=	Filled-In-Ticket + Termination-Address + Time
Delivery-Time-Report	=	*summary of conveyor service*
	=	{Hanger-Number + Conveyor-Time} + Average-Conveyor-Time
Diagn.	=	*IRIS-50 system diagnostics and dumps*
Dock-Sched.	=	*retailers' loading times*
	=	Dock-Number + 1{Buyer-ID + Time}
Filled-In-Ticket	=	*base information on any ticket after sale*
	=	Hanger-Ticket + Buyer-ID + Negotiated-Amount
Hanger-Ticket	=	*document attached to a lot at time of receipt*
	=	Hanger-Number + Wholesaler-ID + Qty + Item-Description
Hangers-In-Stock-Line	=	*report detail on leftover stock*
	=	Hanger-Number + 1{Qty + Item-Description}

Hold	=	*transaction to allocate funds*
Inventory-Status	=	*report on stock movement*
	=	Date + {Arrival-Line} + {Delivery-Line} + {Unclaimed-Goods-Line} + {Hangers-In-Stock-Line}
Invoice	=	Buyer-Statement
Mail-Payment	=	*remittance for previous bill*
	=	[Invoice | Invoice-Number] + Buyer-ID + Check
Market-Sales-Summary	=	*report to tax authority*
	=	Market-Tax-ID + Gross-Sales + {Tax-Category + Amount-Collected} + Total-Tax-Due
Market-Tax-Summary	=	*declaration of TVA withheld by wholesaler*
	=	Market-Tax-ID + Gross-Sales + {Wholesaler-ID + Tax-Category + Amount-Collected}
Order-Form	=	*documentation of real or tentative sale*
	=	Wholesaler-ID + [Buyer-ID | Buyer-Name] + 1{Qty + Item-Description + Amount} + Total-Amount
Order-Packet	=	*working document of sale*
	=	Order-Form + 1{Filled-In-Ticket}
Order	=	*tentative offer for goods*
	=	[Buyer-Name | Buyer-ID] + 1{Qty + Item-Description}
Outgoing-Sale	=	*goods on their way to retailer*
	=	Hanger + Termination-Address + Delivery-Ticket
Payment	=	*transaction to record remittance*
Pickup-Ticket	=	*notification to collect goods*
	=	Buyer-ID + Time + Dock-Number
Receipts-File	=	*record of arrivals correlated to hanger*
	=	Delivery-Document + Hanger-Number + 1{Qty + Item-Description}
Red-Tickets	=	*markers for sold items*
	=	1{Filled-In-Ticket}
Registr.	=	*arriving buyer checkin*
	=	Buyer-Name + Mailing-Address + (Invoice + Check) + (Deposit)
Release	=	*trucker's receipt for accepted lot*
	=	Delivery-Document + Signature
Retail-Receipt	=	*acknowledgement of received goods*
	=	Hanger-Ticket + Signature

Sale	=	*transaction to record completed order*
Sales-Area	=	*hangers of sold and unsold goods*
	=	{Hanger + ([Hanger-Ticket | Filled-In-Ticket | Delivery-Document])}
Sales-Receipt	=	*buyer's copy of negotiatd transaction*
	=	Buyer-ID + Order-Number + Amount-of-Sale + 1{Yellow-Ticket}
Send-List	=	*dispatching instructions*
	=	Wholesaler-ID + 1{Order-Number + Termination-Address + Time}
Sent-Goods-Ticket	=	*documentation of dispatched hanger*
	=	Filled-In-Ticket + Time
Sold-Order	=	*sale documents for dispatching*
	=	Completed-Order
Stock	=	*arriving goods for sale during the day*
	=	Delivery-Document + 1{Commodity}
Summary-of-Charges	=	*costs allocated to wholesaler*
	=	Wholesaler-ID + Date + Floor-Space-Charge + {Dock-Number + Buyer-ID + Time} Total-Dock-Space-Charge
Tax-Package	=	*sales tax for day's activities*
	=	Market-Sales-Summary + Market-Tax-Summary + Check
Ticketed-Hanger	=	*goods ready for sale*
	=	Hanger + Hanger-Ticket
Tkt-Copies	=	*ticket sets documeting completed sales*
	=	{Hanger-Ticket}
Tkt-Sets	=	*ticket sets documenting delivered orders
	=	{Hanger-Ticket + Red-Ticket + Retail-Receipt}
Unclaimed-Goods-Line	=	*report detail on incomplete sales*
	=	Hanger-Number + Buyer-ID
Usage-Report	=	*dock time to be billed to wholesalers*
	=	{Wholesaler-ID + Dock-Number + Total-Elapsed-Time}
Use-Card	=	*dock time charged to wholesaler*
	=	Dock-Number + (Buyer-ID) + 1{Wholesaler-ID} + Time-Used
Used-Sets	=	Tkt-Sets

Wholesale-Lot	=	*goods dispatched to wholesale booth*
	=	Delivery-Document + Hanger + Termination-Address
Wholesaler-Report	=	*summary of day's sales*
	=	Wholesaler-Statement + Wholesaler-Summary
Whlesaler-Resolution	=	*summary and payment of day's receipts*
	=	Wholesaler-Report + Summary-of-Charges + Payment-Due + Check
Yellow-Ticket	=	*buyer's copy of negotiated transaction*
	=	Filled-In-Ticket

APPENDIX VI

DATA FLOW DIAGRAMS

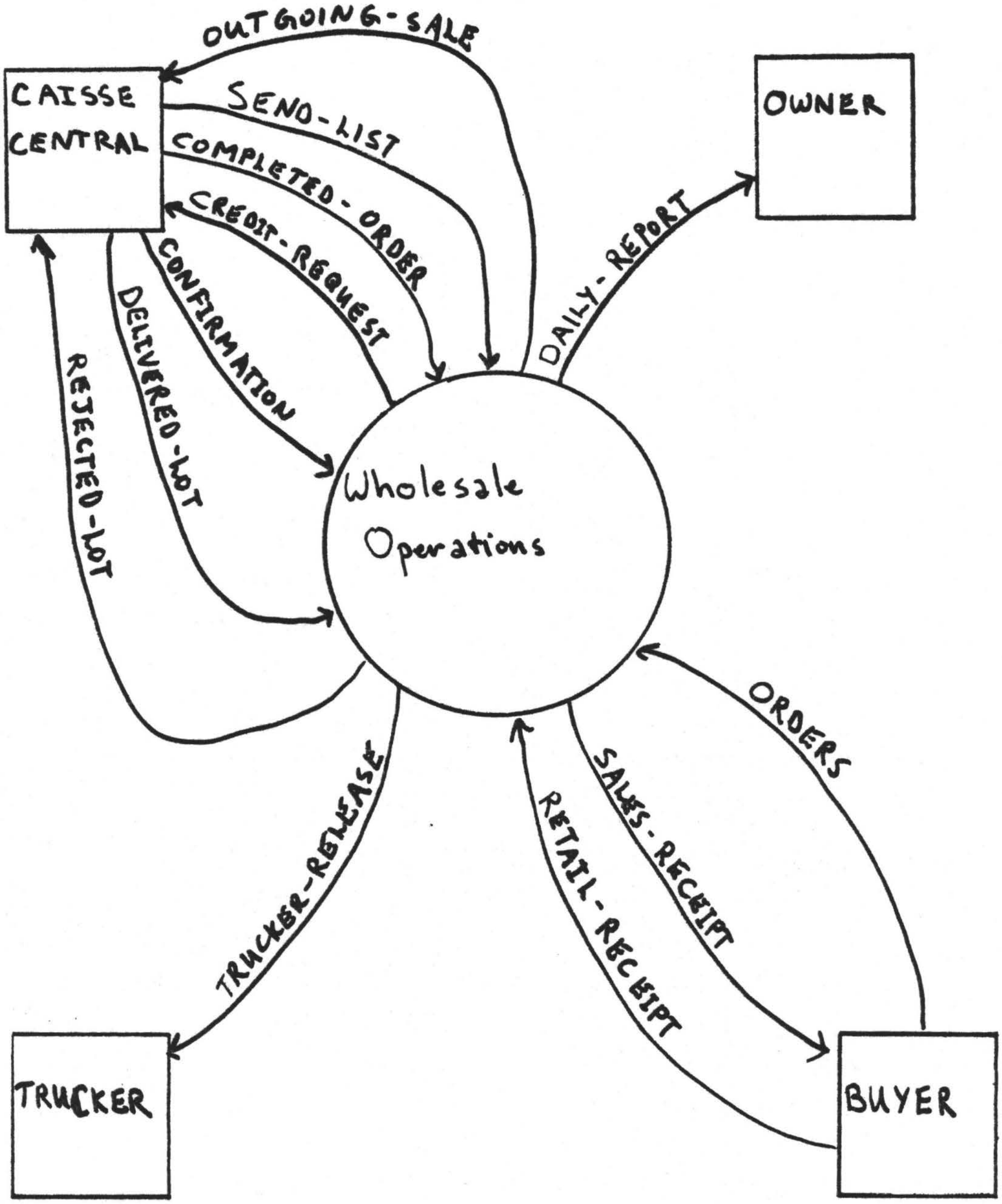
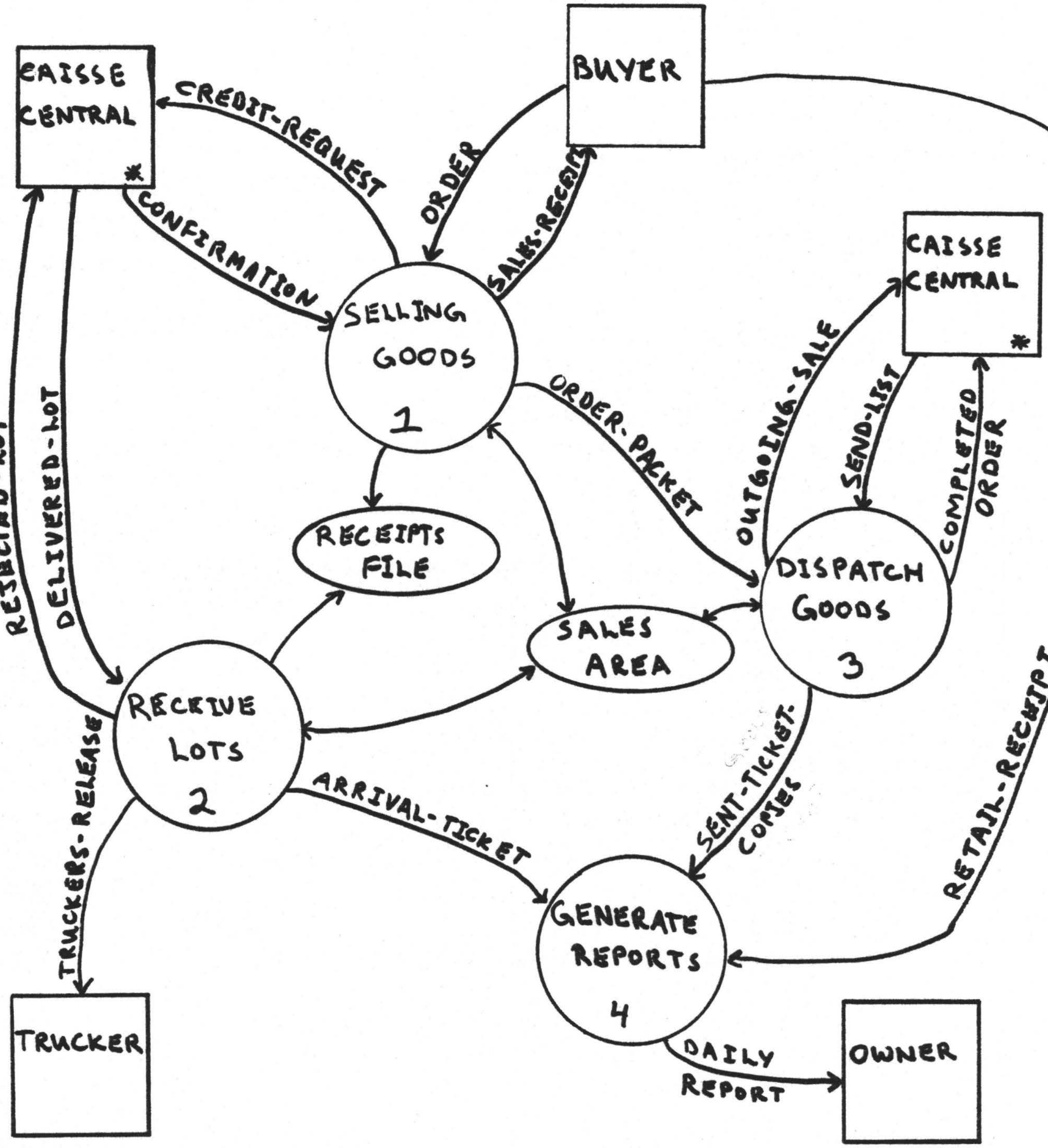

OUTGOING-SALE
CAISSE CENTRAL
SEND-LIST
COMPLETED-ORDER
CREDIT-REQUEST
CONFIRMATION
DELIVERED-LOT
REJECTED-LOT
OWNER
DAILY-REPORT
Wholesale Operations
ORDERS
SALES-RECEIPT
RETAIL-RECEIPT
TRUCKER-RELEASE
TRUCKER
BUYER

DIAGRAM 0



CAISSE CENTRAL
BUYER
CREDIT-REQUEST
ORDER
SALES-RECEIPT
CONFIRMATION
SELLING GOODS 1
CAISSE CENTRAL
OUTGOING-SALE
SEND-LIST
COMPLETED ORDER
ORDER-PACKET
RECEIPTS FILE
REJECTED-LOT
DELIVERED-LOT
DISPATCH GOODS 3
SALES AREA
RECEIVE LOTS 2
TRUCKERS-RELEASE
ARRIVAL-TICKET
SENT-TICKET-COPIES
RETAIL-RECEIPT
GENERATE REPORTS 4
DAILY REPORT
TRUCKER
OWNER

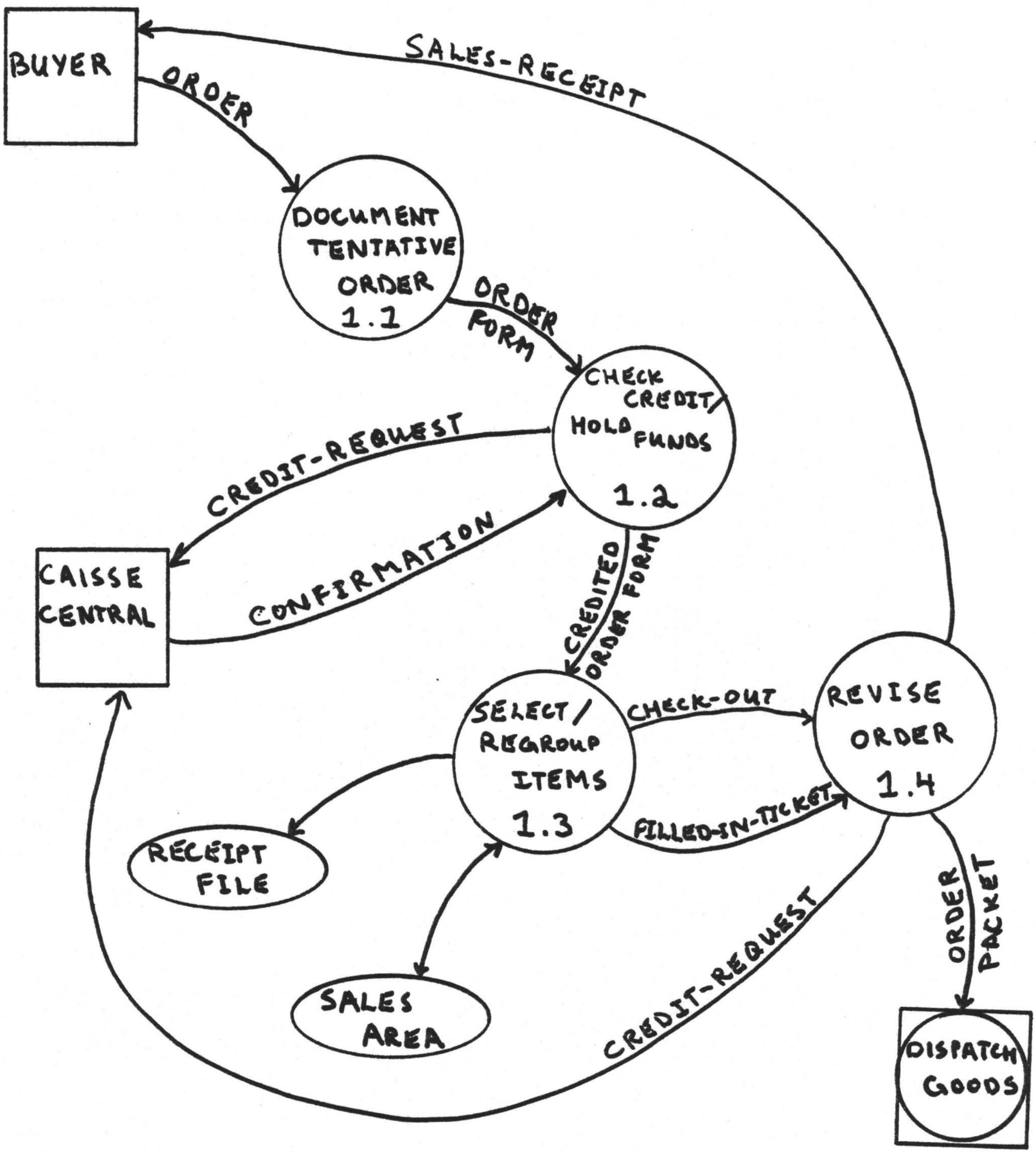
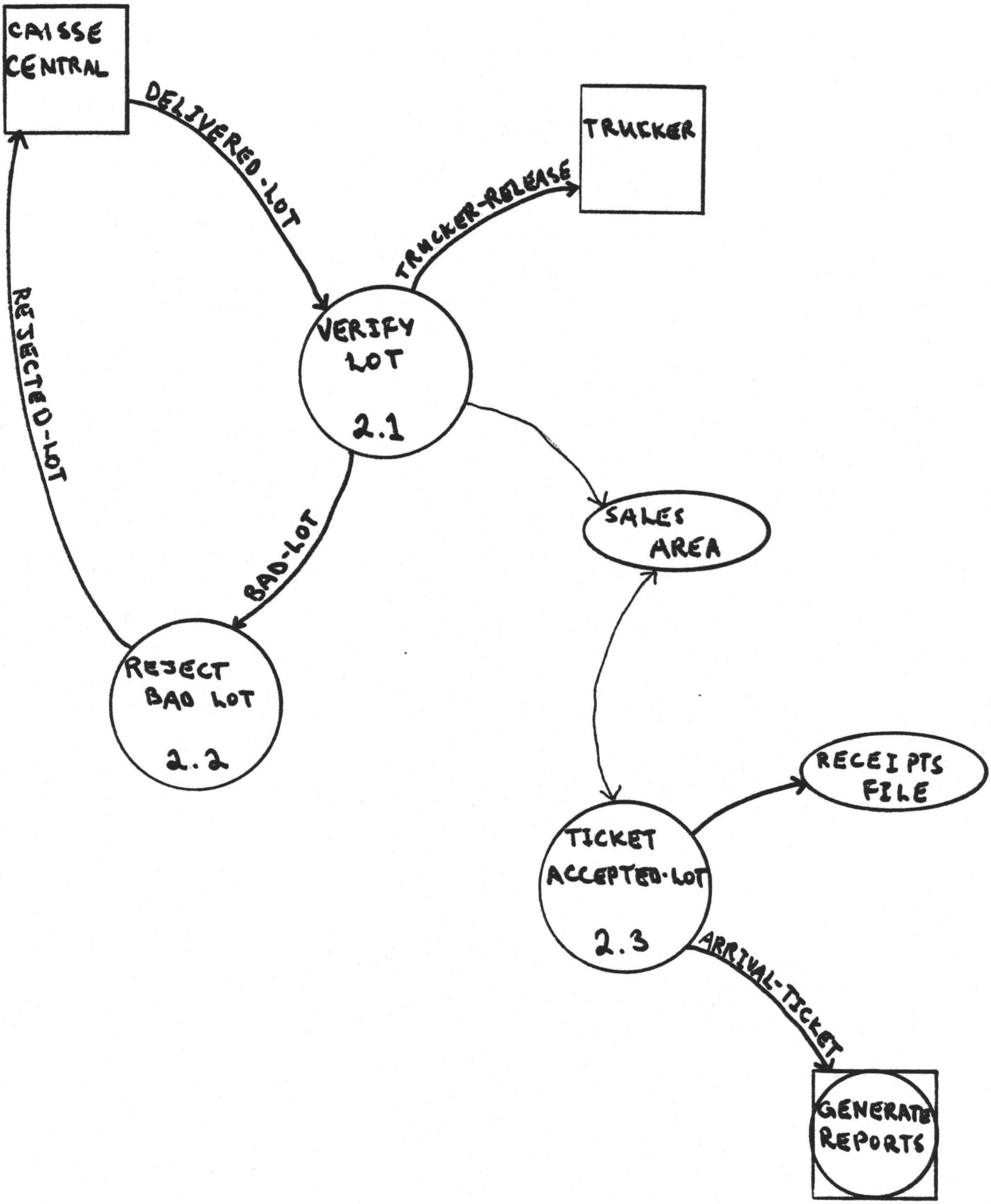
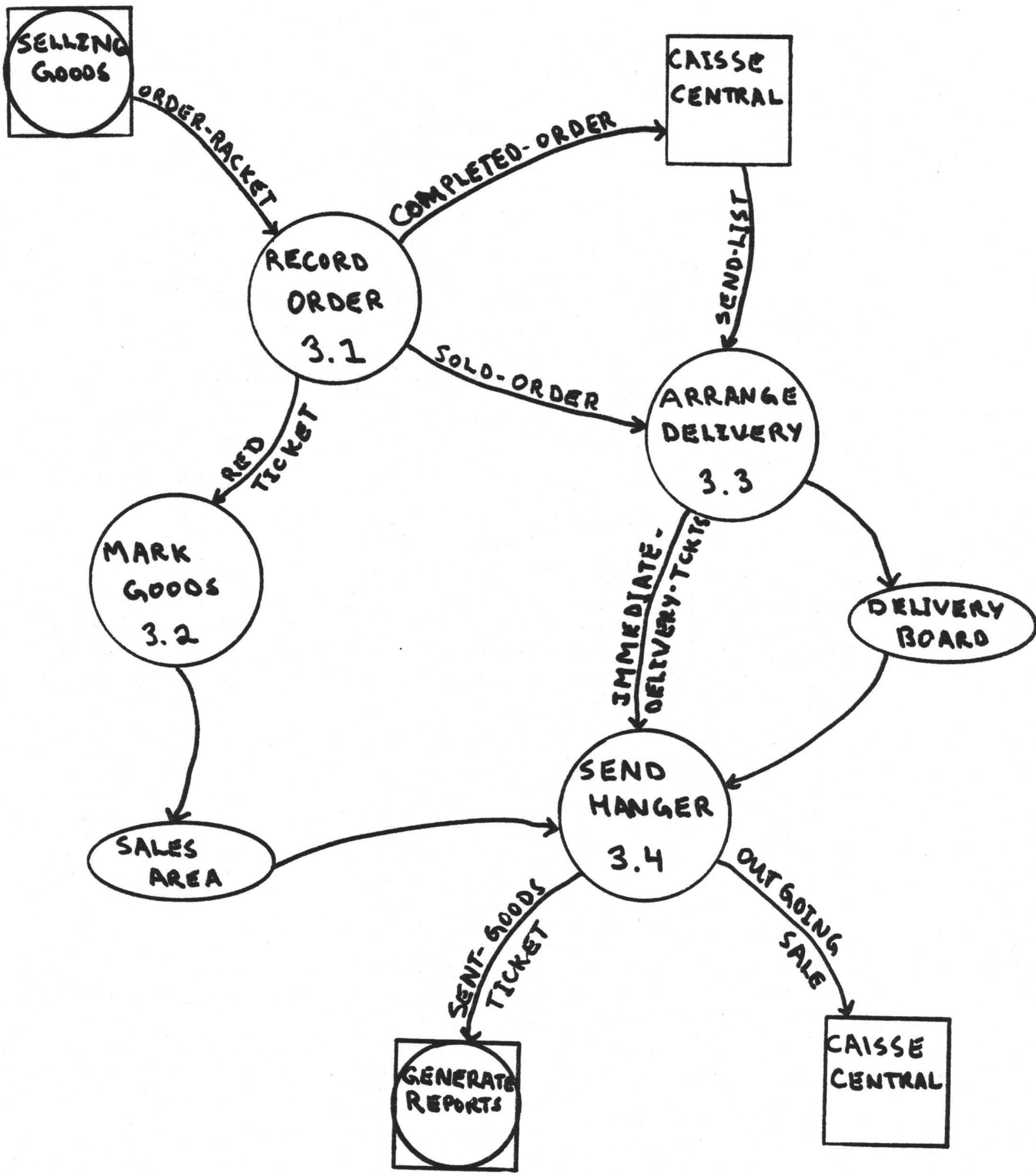
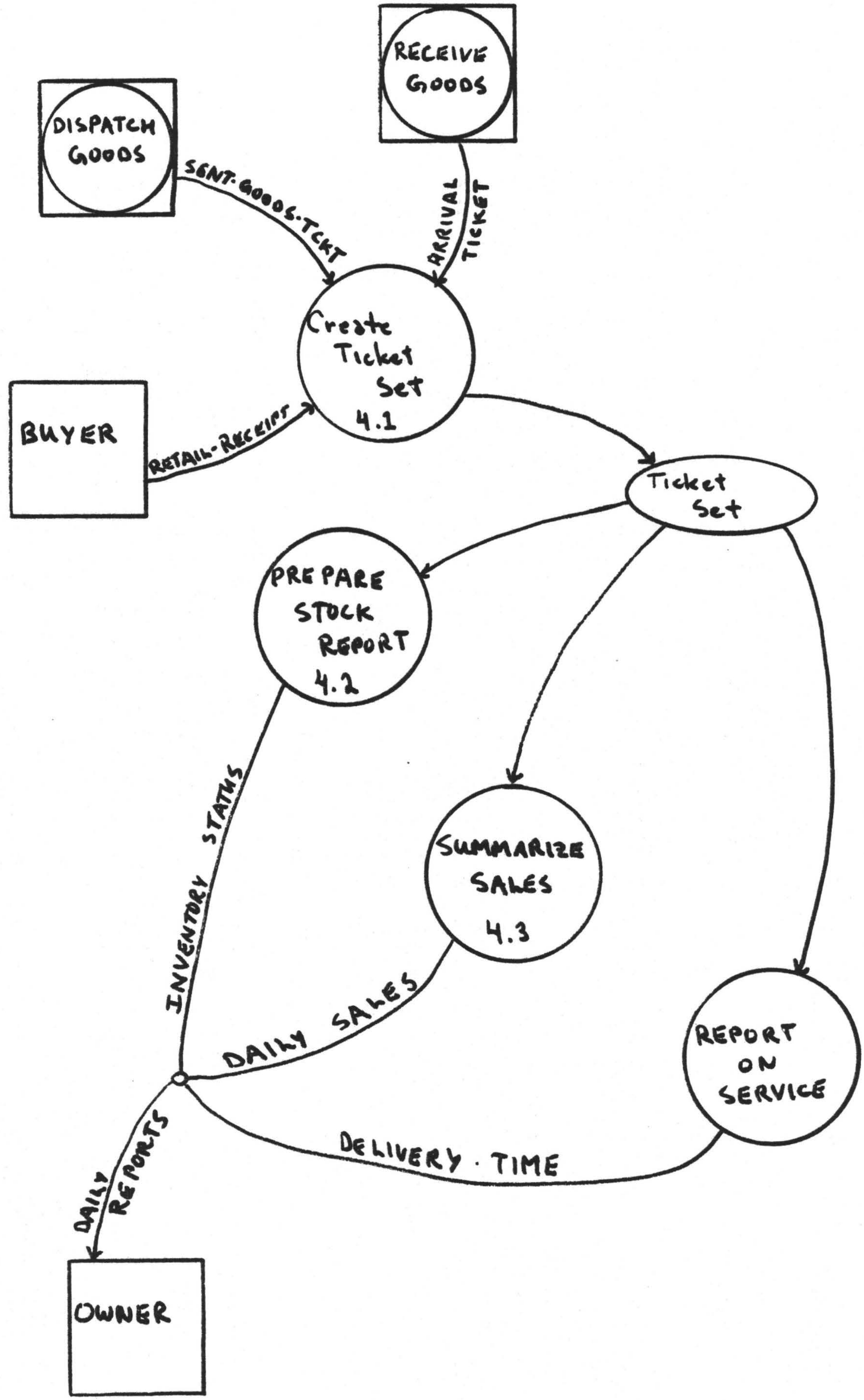

BUYER
SALES-RECEIPT
ORDER
DOCUMENT TENTATIVE ORDER 1.1
ORDER FORM
CHECK CREDIT/ HOLD FUNDS 1.2
CREDIT-REQUEST
CONFIRMATION
CAISSE CENTRAL
CREDITED ORDER FORM
SELECT/ REGROUP ITEMS 1.3
CHECK-OUT
FILLED-IN-TICKET
REVISE ORDER 1.4
RECEIPT FILE
SALES AREA
CREDIT-REQUEST
ORDER PACKET
DISPATCH GOODS

DIAGRAM 2


CAISSE CENTRAL
DELIVERED-LOT
TRUCKER
TRUCKER-RELEASE
REJECTED-LOT
VERIFY LOT
2.1
SALES AREA
BAD-LOT
REJECT BAD LOT
2.2
RECEIPTS FILE
TICKET ACCEPTED-LOT
2.3
ARRIVAL-TICKET
GENERATE REPORTS


SELLING GOODS
ORDER-RACKET
CAISSE CENTRAL
COMPLETED-ORDER
RECORD ORDER
3.1
SEND-LIST
SOLD-ORDER
ARRANGE DELIVERY
3.3
RED TICKET
MARK GOODS
3.2
IMMEDIATE-DELIVERY-TCKTS
DELIVERY BOARD
SEND HANGER
3.4
SALES AREA
SENT-GOODS TICKET
OUTGOING SALE
GENERATE REPORTS
CAISSE CENTRAL



DISPATCH GOODS
RECEIVE GOODS
SENT-GOODS-TCKT
ARRIVAL TICKET
Create Ticket Set 4.1
BUYER
RETAIL-RECEIPT
Ticket Set
PREPARE STOCK REPORT 4.2
SUMMARIZE SALES 4.3
REPORT ON SERVICE
INVENTORY STATUS
DAILY SALES
DELIVERY · TIME
DAILY REPORTS
OWNER

APPENDIX VII

PSA - STRUCTURED ANALYSIS REPORTS

List of Sample Reports

FROMAGE-DU-MIDI

PICTURE REPORT

NAME=WHOLESALE-OPERATIONS

```
                                                                                  +--OUTPUT---+
                                                                                  IDAILY-     I
                                                                            ......IREPORTS    I
                                                                            .     I           I
                                                                            .     +-GENERATES-+
+---INPUT---+                                                               .
I           I                                                               .     +--OUTPUT---+
I   ORDER   I.....                                                          .     ISALES-     I
I           I    .                                                          ......IRECEIPT    I
+-RECEIVES--+    .                                                          .     I           I
                 .                                                          .     +-GENERATES-+
+---INPUT---+    .                                                          .
IDELIVERED- I    .                                                          .     +--OUTPUT---+
ILOT        I.....                                                          .     IOUTGOING-  I
I           I    .                                                          ......ISALE       I
+-RECEIVES--+    .                                                          .     I           I
                 .                                                          .     +-GENERATES-+
+---INPUT---+    .                +--PROCESS--+                             .
ICONFIRMATI-I    .                IWHOLESALE- I                             .     +--OUTPUT---+
ION         I.........................IOPERATIONS I.................................ITRUCKERS-  I
I           I    .                I           I                             .     IRELEASE    I
+-RECEIVES--+    .                +-----------+                             .     I           I
                 .                      .                                   .     +-GENERATES-+
+---INPUT---+    .                      .                                   .
IRETAIL-    I    .                      .                                   .     +--OUTPUT---+
IRECEIPT    I.....                      .                                   .     IREJECTED-  I
I           I    .                      .                                   ......ILOT        I
+-RECEIVES--+    .                      .                                   .     I           I
                 .                      .                                   .     +-GENERATES-+
+---INPUT---+    .                      .                                   .
I           I    .                      .                                   .     +--OUTPUT---+
I SEND-LIST I.....                      .                                   .     ICOMPLETED- I
I           I                           .                                   ......IORDER      I
+-RECEIVES--+                           .                                   .     I           I
                                        .                                   .     +-GENERATES-+
                                        .                                   .
                                        .                                   .     +--OUTPUT---+
                                        .                                   .     ICREDIT-    I
                 ...............................................            ......IREQUEST    I
                 .            .             .              .                      I           I
                 .            .             .              .                      +-GENERATES-+
      +--PROCESS--+ +--PROCESS--+ +--PROCESS--+ +--PROCESS--+
      ISELLING-   I I           I IGENERATE-  I IDISPATCH-  I
      IGOODS      I IRECEIVE-LOTI IREPORTS    I IGOODS      I
      I           I I           I I           I I           I
      +-SUBPARTS--+ +-SUBPARTS--+ +-SUBPARTS--+ +-SUBPARTS--+
```

FROMAGE-DU-MIDI

PICTURE REPORT

NAME=RECEIVE-LOT

```
                                          +--PROCESS--+
                                          IWHOLESALE- I
                                          IOPERATIONS I
                                          I           I
                                          +---PART----+
                                                .
                                                .                                           +--OUTPUT---+
                                                .                                           ITRUCKERS-  I
                                                .                                     ......IRELEASE    I
                                                .                                     .     I           I
                                                .                                     .     +-GENERATES-+
                                                .                                     .     +--OUTPUT---+
                                                .                                     .     IREJECTED-  I
                                                .                                     ......ILOT        I
                                                .                                     .     I           I
                                                .                                     .     +-GENERATES-+
                                                .                                     .     +--ENTITY---+
                                                .                                     .     IHANGER-    I
                                                .                                     ......ITICKET     I
                                                .                                     .     I           I
                                                .                                     .     +--DERIVES--+
+---INPUT---+                             +--PROCESS--+                               .     +--OUTPUT---+
IDELIVERED- I                             I           I                               .     ITRUCKERS-  I
ILOT        I.............................IRECEIVE-LOTI...............................:.....IRELEASE    I
I           I                             I           I                               .     I           I
+-RECEIVES--+                             +-----------+                               .     +--DERIVES--+
                                                .                                     .     +--OUTPUT---+
                                                .                                     .     IREJECTED-  I
                                                .                                     ......ILOT        I
                                                .                                     .     I           I
                                                .                                     .     +--DERIVES--+
                                                .                                     .     +----SET----+
                                                .                                     .     IRECEIPTS-  I
                                                .                                     ......IFILE       I
                                                .                                     .     I           I
                                                .                                     .     +--UPDATES--+
                                                .                                     .     +----SET----+
                                                .                                     .     I           I
                                                .                                     ......ISALES-AREA I
                          ......................................                            I           I
                          .          .          .          .                                +--UPDATES--+
                          .          .          .          .
+---INPUT---+ +--PROCESS--+ +--PROCESS--+ +--PROCESS--+
IDELIVERED- I I           I IREJECT-    I ITICKET-    I
ILOT        I IVERIFY-LOT I IBAD-       I IACCEPTED-  I
I           I I           I ILOT        I ILOT        I
+--EMPLOYS--+ +-SUBPARTS--+ +-SUBPARTS--+ +-SUBPARTS--+
```

PICTURE REPORT

NAME=SELECT-AND-REGROUP-ITEMS

```
                              +--PROCESS--+
                              ISELLING-   I
                              IGOODS      I
                              I           I
                              +---PART----+
                                    .
                                    .
                                    .
+--ENTITY---+                                                    +--ENTITY---+
I           I                                                    IFILLED-    I
ITICKET-COPYI.....                                          .....IIN-        I
I           I    .        +--PROCESS--+                     .    ITICKETS    I
+USES TO DRV+    .        ISELECT-AND-I                     .    +--DERIVES--+
+--ENTITY---+    .........IREGROUP-   I.....................
ICREDITED-  I    .        IITEMS      I                     .    +--ENTITY---+
IORDER-     I.....        +-----------+                     .....I CHECK-OUT I
IFORM       I                   .                                I           I
+USES TO DRV+                   .                                +--DERIVES--+
                                .
                          ..............
                          .            .
                +--ENTITY---+ +--ENTITY---+
                I           I I           I
                ITICKET-COPYI I  HANGER   I
                I           I I           I
                +--REMOVES--+ +-MODIFIES--+
```

144

FROMAGE-DU-MIDI

FUNCTION FLOW DATA DIAGRAM

PARAMETERS: DB=CHEESE.DB FILE=PSATEMP.NAME NOTOP-ATTRIBUTE NOBOTTOM-ATTRIBUTE NONODE-ATTRIBUTE MISSING-ATTRIBUTE-VALUE='N/A' BOX-CHARACTER='I-++++' SYNONYM NOUPDATED-AS-INPUT NODATA-OUT-DASHES NOINDEX NOPUNCHED-NAMES NONEW-PAGE EXPLANATION DATA-IN-ORDER=STANDARD DATA-OUT-ORDER=STANDARD NOPLOT

THIS REPORT PRESENTS, IN GRAPHICAL FORM, THE DATA ENTERING OR LEAVING A PROCESS OR INTERFACE. THE PROCESS OR INTERFACE IN QUESTION IS REPRESENTED AS A BOX CENTERED IN THE PAGE. A SYNONYM AND A VALUE FOR A USER-SPECIFIED ATTRIBUTE ("NODE-ATTRIBUTE") MAY APPEAR IN THE BOX BELOW THE BASIC NAME OF THE PROCESS OR INTERFACE.

DATA OBJECTS ENTERING (DISPLAYED ON THE LEFT OR ABOVE THE BOX, DEPENDING UPON PAGE WIDTH) ARE DETERMINED FROM THE RECEIVES, EMPLOYS, USED TO DERIVE, USED TO UPDATE, AND OPTIONALLY UPDATES RELATIONSHIPS.

DATA OBJECTS LEAVING (DISPLAYED ON THE RIGHT OR BELOW THE BOX, DEPENDING UPON PAGE WIDTH) ARE DETERMINED FROM THE DERIVES, GENERATES, AND UPDATES RELATIONSHIPS.

ATTRIBUTE VALUES OF THE PROCESS OR INTERFACE MAY ALSO BE DISPLAYED AS THE TOP OR BOTTOM SECTIONS OF THE BOX.

THE HEIGHT OF THE BOX IS SET BY THE GREATER OF THE NUMBERS OF DATA OBJECTS ENTERING AND LEAVING THE PROCESS OR INTERFACE.

```
                                 +--------------------------------+
RETAIL-RECEIPT ------------------I                                I- TICKET-SET
HANGER-TICKET -------------------I       CREATE-TICKET-SET        I
SENT-GOODS-TICKETS --------------I                                I
                                 +--------------------------------+


                                 +--------------------------------+
ORDER ---------------------------I    DOCUMENT-TENTATIVE-ORDER    I- ORDER-FORM
                                 I                                I
                                 +--------------------------------+


                                 +--------------------------------+
ORDER ---------------------------I                                I- CREDIT-REQUEST
CONFIRMATION --------------------I                                I- SALES-RECEIPT
                                 I         SELLING-GOODS          I- ORDER-PACKET
                                 I                                I- RECEIPTS-FILE
                                 I                                I- SALES-AREA
                                 +--------------------------------+


                                 +--------------------------------+
DELIVERED-LOT -------------------I                                I- ACCEPTED-LOT
                                 I           VERIFY-LOT           I- BAD-LOT
                                 I                                I- TRUCKERS-RELEASE
                                 +--------------------------------+
```

FROMAGE-DU-MIDI

FORMATTED PROBLEM STATEMENT

```
PARAMETERS:   DB=CHEESE.DB  FILE=FPS.FILE  NOINDEX  NOPUNCHED-NAMES  PRINT  NOPUNCH  SMARG=5
  NMARG=20  AMARG=10  BMARG=25  RNMARG=72  CMARG=1  HMARG=40  ONE-PER-LINE  COMMENT  NONEW-PAGE
  NONEW-LINE  NOALL-STATEMENTS  COMPLEMENTARY-STATEMENTS  LINE-NUMBERS  PRINTEOF  DLC-COMMENT
  NOSORT-NAME-LIST

DEFINE ELEMENT                              HANGER-NUMBER;
     /*   DATE OF LAST CHANGE - DEC 09, 1980, 10:12:18 */
     DESCRIPTION;
 THIS IS THE NUMBER ASSIGNED TO THE HANGER AT THE TIME IT ARRIVES.;
     CONTAINED IN:  ARRIVAL-RECEIPT,
                    SERVICE-TIMES,
                    ARRIVAL-LINE,
                    DELIVERY-LINE,
                    UNCLAIMED-GOODS-LINE,
                    HANGER-IN-STOCK-LINE,
                    HANGER-TICKET-FORM;
     USED BY:
          TICKET-ACCEPTED-LOT
           TO DERIVE      ARRIVAL-RECEIPT;

DEFINE ENTITY                               CHECK-OUT;
     /*   DATE OF LAST CHANGE - DEC 09, 1980, 10:12:18 */
     DESCRIPTION;
THIS IS THE DEPARTING CUSTOMERS NOTICE OF COMPLETION.;
     CONSISTS OF:
          SALESMAN-APPROVAL,
          BUYER-NAME,
          ZERO-OR-MORE    CREDITED-ORDER-GROUP;
     DERIVED BY:     SELECT-AND-REGROUP-ITEMS
       USING:        CREDITED-ORDER-FORM;
     USED BY:
          REVISE-ORDER
           TO DERIVE      CREDIT-REQUEST;
     USED BY:
          REVISE-ORDER
           TO DERIVE      REVISED-ORDER;

DEFINE GROUP                                GREEN-TICKET-COPY;
     /*   DATE OF LAST CHANGE - NOV 24, 1980, 14:28:02 */
     CONSISTS OF:
          HANGER-TICKET-FORM,
          BUYER-ID,
          NEGOTIATED-AMOUNT;
     CONTAINED IN:  FILLED-IN-TICKETS,
                    REMAINING-TICKET-COPIES,
                    ONE-OR-MORE
                     TIMES IN COMPLETED-ORDER,
                    FILLED-IN-TICKET-GROUP;
     DERIVED BY:    RECORD-ORDER
       USING:       REMAINING-TICKET-COPIES;
     USED BY:
          RECORD-ORDER
           TO DERIVE      COMPLETED-ORDER;

DEFINE INPUT                                DELIVERED-LOT;
```

146

FORMATTED PROBLEM STATEMENT

```
      /*    DATE OF LAST CHANGE - NOV 24, 1980, 14:28:02 */
      DESCRIPTION;
  THESE ARE THE GOODS ARRIVING AT THE WHOLESALERS BOOTH.;
      GENERATED BY:   CAISSE-CENTRAL;
      RECEIVED BY:    WHOLESALE-OPERATIONS,
                      RECEIVE-LOT,
                      VERIFY-LOT;
      CONSISTS OF:
            HANGER-GROUP,
            DELIVERY-DOCUMENT-FORM;
      USED BY:
            VERIFY-LOT
             TO DERIVE      ACCEPTED-LOT;
      USED BY:
            VERIFY-LOT
             TO DERIVE      BAD-LOT;
      USED BY:
            VERIFY-LOT
             TO DERIVE      TRUCKERS-RELEASE;
      EMPLOYED BY:    RECEIVE-LOT;

 DEFINE INTERFACE                           CAISSE-CENTRAL;
      /*    DATE OF LAST CHANGE - NOV 11, 1980, 11:15:22 */
      GENERATES:
            DELIVERED-LOT,
            CONFIRMATION,
            SEND-LIST;
      RECEIVES:
            OUTGOING-SALE,
            REJECTED-LOT,
            COMPLETED-ORDER,
            CREDIT-REQUEST;

 DEFINE OUTPUT                              DELIVERY-TIME-REPORT;
      /*    DATE OF LAST CHANGE - DEC 05, 1980, 13:44:15 */
      DESCRIPTION;
  THIS IS A SUMMARY OF CONVEYOR SERVICE;
      GENERATED BY:   REPORT-ON-SERVICE;
      PART OF:        DAILY-REPORTS;
      CONSISTS OF:
            ZERO-OR-MORE    SERVICE-TIMES,
            AVERAGE-CONVEYOR-TIME;
      DERIVED BY:     REPORT-ON-SERVICE
        USING:        TICKET-SET;

 DEFINE PROCESS                             RECEIVE-LOT;
      /*    DATE OF LAST CHANGE - NOV 20, 1980, 13:51:26 */
      ATTRIBUTES ARE:
            DIAGRAM          D-0,
            PROCESS-NUMBER   P-2;
      GENERATES:
            TRUCKERS-RELEASE,
            REJECTED-LOT;
      RECEIVES:
            DELIVERED-LOT;
```

FORMATTED PROBLEM STATEMENT

```
        SUBPARTS ARE:   VERIFY-LOT,
                        REJECT-BAD-LOT,
                        TICKET-ACCEPTED-LOT;
        PART OF:        WHOLESALE-OPERATIONS;
        DERIVES:        HANGER-TICKET;
        DERIVES:        TRUCKERS-RELEASE;
        DERIVES:        REJECTED-LOT;
        UPDATES:        RECEIPTS-FILE;
        UPDATES:        SALES-AREA;
        EMPLOYS:        DELIVERED-LOT;

    DEFINE PROCESS                              REVISE-ORDER;
        /*   DATE OF LAST CHANGE - DEC 05, 1980, 13:44:15 */
        ATTRIBUTES ARE:
             DIAGRAM            D-1,
             PROCESS-NUMBER P-4;
        GENERATES:
             CREDIT-REQUEST,
             SALES-RECEIPT;
        PART OF:        SELLING-GOODS;
        DERIVES:        CREDIT-REQUEST
          USING:        CHECK-OUT;
        DERIVES:        REVISED-ORDER
          USING:        FILLED-IN-TICKETS,
                        CHECK-OUT;
        DERIVES:        SALES-RECEIPT
          USING:        FILLED-IN-TICKETS,
                        REVISED-ORDER;
        DERIVES:        REMAINING-TICKET-COPIES
          USING:        FILLED-IN-TICKETS,
                        REVISED-ORDER;
        DERIVES:        ORDER-PACKET
          USING:        REMAINING-TICKET-COPIES,
                        REVISED-ORDER;
        PROCEDURE;
     FOR EACH BUYER CHECKING OUT:
          1.  IF ORDER FORM MARKED 'NO SALE'
                    CALL CAISSE TO RELEASE BUYER'S FUNDS
              OTHERWISE
                    COPY ACTUAL AMOUNTS FROM HANGER TICKET ONTO ORDER FORM
                    CALL CAISSE TO CONFIRM ACTUAL AMOUNT OF SALE
                    WRITE SALES RECEIPT FOR BUYER
                    GIVE BUYER YELLOW TICKET COPY OF HANGER TICKET
                    REMAINING TICKETS AND REVISED ORDER COMBINED TO FORM
                          ORDER PACKET.;

    DEFINE SET                                  SALES-AREA;
        /*   DATE OF LAST CHANGE - NOV 26, 1980, 19:12:11 */
        DESCRIPTION;
      THIS IS THE SET OF HANGERS OF SOLD AND UNSOLD GOODS.;
        COLLECTION OF:
             ZERO-OR-MORE     SOLD-AND-UNSOLD-GOODS;
        HAS: ACCEPTED-LOT
             ADDED BY         VERIFY-LOT;
        HAS: HANGER-TICKET
```

FORMATTED PROBLEM STATEMENT

```
            ADDED BY          TICKET-ACCEPTED-LOT;
     HAS: RED-TICKET
            ADDED BY          MARK-GOODS;
     HAS: HANGER
              MODIFIED BY   SELECT-AND-REGROUP-ITEMS;
     HAS: TICKET-COPY
              REMOVED BY    SELECT-AND-REGROUP-ITEMS;
     HAS: DELIVERY-DOCUMENT
              REMOVED BY    TICKET-ACCEPTED-LOT;
     HAS: HANGER,
            RED-TICKET
              REMOVED BY    SEND-HANGER;
     UPDATED BY:     SELLING-GOODS;
     UPDATED BY:     RECEIVE-LOT;

EOF EOF EOF EOF EOF
```

DICTIONARY REPORT

```
PARAMETERS:   DB=CHEESE.DB  FILE=DIC.FILE  NOINDEX  NOPUNCHED-NAMES  DESCRIPTION  SYNONYMS  KEYWORDS
  RESPONSIBLE-PD  ATTRIBUTES  SPACING=2  NONEW-PAGE


1 AMOUNT-OF-SALE                                                                  ELEMENT

        DESCRIPTION:
                   THIS IS THE AMOUNT THE BUYER HAS SPENT.

          SYNONYMS: AMT-OF-SALE                          TOTAL-AMOUNT
                    TOTAL-AMT

2 DOCUMENT-TENTATIVE-ORDER                                                        PROCESS

         ATTRIBUTE:                            VALUE:
                    DIAGRAM                              D-1
                    PROCESS-NUMBER                       P-1

3 IMMED-DELIVERY-TICKET                                                           ENTITY

        DESCRIPTION:
                    THIS IS THE DOCUMENTATION FOR HANGERS WHICH ARE READY TO BE
                    DISPATCHED IMMEDIATELY. THEY ARE THE SAME AS DELIVERY TICKETS BUT
                    HAVE HIGHER PRIORITY.

           SYNONYM: IMMED-DELIVERY-TICKETS
```

FROMAGE-DU-MIDI

CONTENTS REPORT

```
PARAMETERS:   DB=CHEESE.DB  FILE=CONT.FILE  NOCOMPLETENESS-CHECK  NOINDEX  NOPUNCHED-NAMES
  LEVELS=ALL  LINE-NUMBERS  LEVEL-NUMBERS  OBJECT-TYPES  PRINT  NONEW-PAGE

 1* (ENTITY)      1 CHECK-OUT
  1 (ELEMENT)       2   SALESMAN-APPROVAL
  2 (ELEMENT)       2   BUYER-NAME
  3 (GROUP)         2   CREDITED-ORDER-GROUP (ZERO-OR-MORE)
  4 (GROUP)          3      ORDER-FORM-GROUP
  5 (ELEMENT)         4        BLANK-ORDER-FORM
  6 (ELEMENT)         4        WHOLESALER-ID
  7 (ELEMENT)         4        AMOUNT-OF-SALE
  8 (GROUP)           4        ITEM-AND-PRICE (ONE-OR-MORE)
  9 (ELEMENT)          5          QUANTITY
 10 (ELEMENT)          5          ITEM-DESCRIPTION
 11 (ELEMENT)          5          AMOUNT
 12 (GROUP)           4        IDENTIFICATIONS (ONE-OF-THE-FOLLOWING)
 13 (ELEMENT)          5          BUYER-NAME
 14 (ELEMENT)          5          BUYER-ID
 15 (ELEMENT)        3      AMOUNT-HELD
 16 (ELEMENT)        3      MANAGER-APPROVAL
 2* (INPUT)       1 ORDER
  1 (GROUP)         2   ITEMS (ONE-OR-MORE)
  2 (ELEMENT)        3      QUANTITY
  3 (ELEMENT)        3      ITEM-DESCRIPTION
  4 (GROUP)         2   IDENTIFICATIONS (ONE-OF-THE-FOLLOWING)
  5 (ELEMENT)        3      BUYER-NAME
  6 (ELEMENT)        3      BUYER-ID
 3* (OUTPUT)      1 INVENTORY-STATUS-REPORT
  1 (ELEMENT)       2   DATE
  2 (GROUP)         2   ARRIVAL-LINE (ZERO-OR-MORE)
  3 (ELEMENT)        3      HANGER-NUMBER
  4 (GROUP)          3      ITEMS (ONE-OR-MORE)
  5 (ELEMENT)         4        QUANTITY
  6 (ELEMENT)         4        ITEM-DESCRIPTION
  7 (GROUP)         2   DELIVERY-LINE (ZERO-OR-MORE)
  8 (ELEMENT)        3      HANGER-NUMBER
  9 (ELEMENT)        3      BUYER-ID
 10 (GROUP)         2   UNCLAIMED-GOODS-LINE (ZERO-OR-MORE)
 11 (ELEMENT)        3      HANGER-NUMBER
 12 (ELEMENT)        3      BUYER-ID
 13 (GROUP)         2   HANGER-IN-STOCK-LINE (ZERO-OR-MORE)
 14 (ELEMENT)        3      HANGER-NUMBER
 15 (GROUP)          3      ITEMS (ONE-OR-MORE)
 16 (ELEMENT)         4        QUANTITY
 17 (ELEMENT)         4        ITEM-DESCRIPTION
 4* (SET)         1 SALES-AREA
  1 (ENTITY)        2   SOLD-AND-UNSOLD-GOODS (ZERO-OR-MORE)
  2 (GROUP)          3      HANGER-GROUP
  3 (GROUP)           4        CHEESE-TYPES (ONE-OR-MORE)
  4 (ELEMENT)          5          CAMEMBERT
  5 (ELEMENT)          5          BRIE
  6 (GROUP)          3      DOCUMENT (OPTIONAL)
  7 (GROUP)           4        TICKETS (ONE-OF-THE-FOLLOWING)
  8 (GROUP)            5          HANGER-TICKET-FORM
  9 (ELEMENT)           6            HANGER-NUMBER
```

FROMAGE-DU-MIDI

CONTENTS REPORT

```
10 (ELEMENT)      6          WHOLESALER-ID
11 (ELEMENT)      6          ITEM-DESCRIPTION
12 (ELEMENT)      6          QUANTITY
13 (GROUP)       5        FILLED-IN-TICKET-GROUP
14 (GROUP)        6          WHITE-TICKET-COPY
15 (GROUP)         7            HANGER-TICKET-FORM
16 (ELEMENT)        8              HANGER-NUMBER
17 (ELEMENT)        8              WHOLESALER-ID
18 (ELEMENT)        8              ITEM-DESCRIPTION
19 (ELEMENT)        8              QUANTITY
20 (ELEMENT)       7            BUYER-ID
21 (ELEMENT)       7            NEGOTIATED-AMOUNT
22 (GROUP)        6          RED-TICKET-COPY
23 (GROUP)         7            HANGER-TICKET-FORM
24 (ELEMENT)        8              HANGER-NUMBER
25 (ELEMENT)        8              WHOLESALER-ID
26 (ELEMENT)        8              ITEM-DESCRIPTION
27 (ELEMENT)        8              QUANTITY
28 (ELEMENT)       7            BUYER-ID
29 (ELEMENT)       7            NEGOTIATED-AMOUNT
30 (GROUP)        6          GREEN-TICKET-COPY
31 (GROUP)         7            HANGER-TICKET-FORM
32 (ELEMENT)        8              HANGER-NUMBER
33 (ELEMENT)        8              WHOLESALER-ID
34 (ELEMENT)        8              ITEM-DESCRIPTION
35 (ELEMENT)        8              QUANTITY
36 (ELEMENT)       7            BUYER-ID
37 (ELEMENT)       7            NEGOTIATED-AMOUNT
38 (GROUP)        6          YELLOW-TICKET-COPY
39 (GROUP)         7            HANGER-TICKET-FORM
40 (ELEMENT)        8              HANGER-NUMBER
41 (ELEMENT)        8              WHOLESALER-ID
42 (ELEMENT)        8              ITEM-DESCRIPTION
43 (ELEMENT)        8              QUANTITY
44 (ELEMENT)       7            BUYER-ID
45 (ELEMENT)       7            NEGOTIATED-AMOUNT
46 (GROUP)       5        DELIVERY-DOCUMENT-FORM
47 (ELEMENT)      6          TRUCKER-ID
48 (ELEMENT)      6          ORIGINATING-DEALER-NAME
49 (GROUP)        6          ITEMS (ONE-OR-MORE)
50 (ELEMENT)       7            QUANTITY
51 (ELEMENT)       7            ITEM-DESCRIPTION
```

FROMAGE-DU-MIDI

STRUCTURE REPORT

```
PARAMETERS:   DB=CHEESE.DB  FILE=CONT.FILE  OBJECTS=SET,INP,OUT,ENT,ELE,GR,PRC
  RELATIONS=SUBP,CSTS,CLTN  PRINT  INDENT=3  NOINDEX  NOPUNCHED-NAMES  LEVELS=ALL  TYPES-MARGIN=60
  RELATIONS-MARGIN=75  LINE-NUMBERS  LEVEL-NUMBERS  STATISTICS  NONEW-PAGE  NORELATIONS-MATRIX
  EXPLANATION  OBJECT-TYPES  RELATION-TYPES  NOUPWARD  DOWNWARD  ROW-ORDER=STANDARD
  COLUMN-ORDER=STANDARD  NOBOTTOM-LEVEL-ONLY  NOTOP-LEVEL-ONLY  NOCOMPRESS-RELATION-MATRIX
```

THE STRUCTURES PRESENTED, FOLLOW A FIXED FORMAT. INPUT NAMES ARE ALWAYS AT LEVEL ONE IN A STRUCTURE. IF THE DOWNWARD(FORWARD) PARAMETER IS SPECIFIED, NAMES RELATED TO THE INPUT NAME ARE AT LEVEL TWO, NAMES RELATED TO THOSE NAMED ARE LEVEL THREE, ETC. IF THE UPWARD(BACKWARD) PARAMETER IS SPECIFIED, NAMES THAT THE INPUT NAME ARE RELATED TO WILL BE LEVEL TWO NAMES, NAMES THAT THOSE NAMES ARE RELATED TO ARE LEVEL THREE NAMES, ETC. THE OBJECT TYPE APPEARS IN THE FIRST OF THE TWO RIGHT SIDE COLUMNS. THE RELATIONSHIP BETWEEN TWO OBJECTS IS DISPLAYED IN THE RIGHT MOST COLUMN. AN EXAMPLE OF THIS FORMAT IS:

```
 1       1 PROCESS-1              PROCESS
 2        2   PROCESS-2           PROCESS      (UTILIZES)
 3        2   PROCESS-3           PROCESS      (SUBPARTS,UTILIZES)
 4         3    PROCESS-4         PROCESS      (UTILIZES)
```

THE INTERPRETATION OF THIS STRUCTURE IS THE FOLLOWING:
THE INPUT NAME IS PROCESS-1. PROCESS-1 UTILIZES PROCESS-2. PROCESS-1 UTILIZES AND IS SUBPART OF PROCESS-3. PROCESS-3 UTILIZES PROCESS-4.
IF UPWARDS IS SPECIFIED INSTEAD OF DOWNWARDS, THE LEVEL TWO AND THREE NAMES IN THE STRUCTURE WOULD REPRESENT HIGHER LEVEL RELATIONS INSTEAD OF LOWER. THE FOLLOWING UPWARDS RELATION TYPES WOULD REPLACE THOSE PRESENTED; (UTILIZED),(PART,UTILIZED), AND (UTILIZED) RESPECTIVELY.

IF BOTH UPWARDS AND DOWNWARDS ARE SPECIFIED, THE DOWNWARDS STRUCTURE WILL BE PRESENTED FIRST, FOLLOWED BY THE UPWARDS STRUCTURE.

```
 1    1 CHECK-OUT                                  ENTITY
 2     2   SALESMAN-APPROVAL                       ELEMENT      (CONSISTS OF)
 3     2   BUYER-NAME                              ELEMENT      (CONSISTS OF)
 4     2   CREDITED-ORDER-GROUP                    GROUP        (CONSISTS OF)
 5      3    ORDER-FORM-GROUP                      GROUP        (CONSISTS OF)
 6       4     BLANK-ORDER-FORM                    ELEMENT      (CONSISTS OF)
 7       4     WHOLESALER-ID                       ELEMENT      (CONSISTS OF)
 8       4     AMOUNT-OF-SALE                      ELEMENT      (CONSISTS OF)
 9       4     ITEM-AND-PRICE                      GROUP        (CONSISTS OF)
10        5      QUANTITY                          ELEMENT      (CONSISTS OF)
11        5      ITEM-DESCRIPTION                  ELEMENT      (CONSISTS OF)
12        5      AMOUNT                            ELEMENT      (CONSISTS OF)
13       4     IDENTIFICATIONS                     GROUP        (CONSISTS OF)
14        5      BUYER-NAME                        ELEMENT      (CONSISTS OF)
15        5      BUYER-ID                          ELEMENT      (CONSISTS OF)
16      3    AMOUNT-HELD                           ELEMENT      (CONSISTS OF)
17      3    MANAGER-APPROVAL                      ELEMENT      (CONSISTS OF)
```

FROMAGE-DU-4191

STRUCTURE REPORT

```
LEVEL COUNT   LEVEL COUNT   LEVEL COUNT   LEVEL COUNT   LEVEL COUNT
    1     1       2     3       3     3       4     5       5     5


  1    1 ORDER                                          INPUT
  2     2   ITEMS                                       GROUP          (CONSISTS OF)
  3      3      QUANTITY                                ELEMENT        (CONSISTS OF)
  4      3      ITEM-DESCRIPTION                        ELEMENT        (CONSISTS OF)
  5     2   IDENTIFICATIONS                             GROUP          (CONSISTS OF)
  6      3      BUYER-NAME                              ELEMENT        (CONSISTS OF)
  7      3      BUYER-ID                                ELEMENT        (CONSISTS OF)


LEVEL COUNT   LEVEL COUNT   LEVEL COUNT
    1     1       2     2       3     4


  1    1 INVENTORY-STATUS-REPORT                        OUTPUT
  2     2   DATE                                        ELEMENT        (CONSISTS OF)
  3     2   ARRIVAL-LINE                                GROUP          (CONSISTS OF)
  4      3      HANGER-NUMBER                           ELEMENT        (CONSISTS OF)
  5      3      ITEMS                                   GROUP          (CONSISTS OF)
  6       4        QUANTITY                             ELEMENT        (CONSISTS OF)
  7       4        ITEM-DESCRIPTION                     ELEMENT        (CONSISTS OF)
  8     2   DELIVERY-LINE                               GROUP          (CONSISTS OF)
  9      3      HANGER-NUMBER                           ELEMENT        (CONSISTS OF)
 10      3      BUYER-ID                                ELEMENT        (CONSISTS OF)
 11     2   UNCLAIMED-GOODS-LINE                        GROUP          (CONSISTS OF)
 12      3      HANGER-NUMBER                           ELEMENT        (CONSISTS OF)
 13      3      BUYER-ID                                ELEMENT        (CONSISTS OF)
 14     2   HANGER-IN-STOCK-LINE                        GROUP          (CONSISTS OF)
 15      3      HANGER-NUMBER                           ELEMENT        (CONSISTS OF)
 16      3      ITEMS                                   GROUP          (CONSISTS OF)
 17       4        QUANTITY                             ELEMENT        (CONSISTS OF)
 18       4        ITEM-DESCRIPTION                     ELEMENT        (CONSISTS OF)


LEVEL COUNT   LEVEL COUNT   LEVEL COUNT   LEVEL COUNT
    1     1       2     5       3     8       4     4
```

FROMAGE-DU-MIDI

STRUCTURE REPORT

1	1	WHOLESALE-OPERATIONS	PROCESS	
2	2	SELLING-GOODS	PROCESS	(SUBPARTS ARE)
3	3	DOCUMENT-TENTATIVE-ORDER	PROCESS	(SUBPARTS ARE)
4	3	CHECK-CREDIT-HOLD-FUNDS	PROCESS	(SUBPARTS ARE)
5	3	SELECT-AND-REGROUP-ITEMS	PROCESS	(SUBPARTS ARE)
6	3	REVISE-ORDER	PROCESS	(SUBPARTS ARE)
7	2	RECEIVE-LOT	PROCESS	(SUBPARTS ARE)
8	3	VERIFY-LOT	PROCESS	(SUBPARTS ARE)
9	3	REJECT-BAD-LOT	PROCESS	(SUBPARTS ARE)
10	3	TICKET-ACCEPTED-LOT	PROCESS	(SUBPARTS ARE)
11	2	GENERATE-REPORTS	PROCESS	(SUBPARTS ARE)
12	3	SUMMARIZE-SALES	PROCESS	(SUBPARTS ARE)
13	3	REPORT-ON-SERVICE	PROCESS	(SUBPARTS ARE)
14	3	PREPARE-STOCK-REPORT	PROCESS	(SUBPARTS ARE)
15	3	CREATE-TICKET-SET	PROCESS	(SUBPARTS ARE)
16	2	DISPATCH-GOODS	PROCESS	(SUBPARTS ARE)
17	3	MARK-GOODS	PROCESS	(SUBPARTS ARE)
18	3	ARRANGE-DELIVERY	PROCESS	(SUBPARTS ARE)
19	3	SEND-HANGER	PROCESS	(SUBPARTS ARE)
20	3	RECORD-ORDER	PROCESS	(SUBPARTS ARE)

LEVEL	COUNT	LEVEL	COUNT	LEVEL	COUNT
1	1	2	4	3	15

FROMAGE-DU-MIDI

DATA BASE SUMMARY

PARAMETERS: DB=CHEESE.DB PERCENT SYNONYM DESCRIPTION NORESPONSIBLE-PROBLEM-DEFINER NOSOURCE
NOSECURITY NOKEYWORD NOATTRIBUTE NOMEMO NOTRACE-KEY NOASSERT NOASSERTED

OBJECT TYPE	COUNT	NUMBER WITH SYNONYM	PERCENT WITH SYNONYM	NUMBER WITH DESC	PERCENT WITH DESC
ATTRIBUTE	2	2	100.00	0	
ELEMENT	30	12	40.00	30	100.00
ENTITY	22	3	13.64	22	100.00
GROUP	30	5	16.67	6	20.00
INPUT	5	0		5	100.00
INTERFACE	4	0		0	
MEMO	1	0		1	100.00
OUTPUT	10	1	10.00	10	100.00
PROCESS	20	0		0	
SET	4	0		4	100.00
SYSTEM-PARAMETER	14	0		0	
** TOTAL **	142	23	16.20	78	54.93

PUNCHED COMMENT ENTRIES

```
PARAMETERS:   DB=CHEESE.DB   FILE=PSATEMP.NAME   NODERIVATION   NODESCRIPTION   NODISTRIBUTION
   NOFALSE-WHILE  NOLAYOUT   PROCEDURE   NOTRUE-WHILE   NOVOLATILITY   NOVOLATILITY-MEMBER
   NOVOLATILITY-SET   PRINT   PUNCH=PSATEMP.PCON   EMPTY   LINE-NUMBERS   NONEW-PAGE

1*   CHECK-CREDIT-HOLD-FUNDS
     PROCEDURE;
      FOR EACH TENTATIVE-ORDER-FORM:
            1.   CALCULATE THE ESTIMATED TOTAL OF PURCHASE.
            2.   CALL CAISSE CENTRAL TO CONFIRM BUYER'S CREDIT.
            3.   IF CONFIRMATION POSITIVE
                       INITIAL ORDER FORM
                       SEND BUYER TO SALES AREA
                 OTHERWISE
                       REFUSE THE BUYER CREDIT.;

2*   DOCUMENT-TENTATIVE-ORDER
     PROCEDURE;
      FOR EACH BUYER:
            1.   GREET THEM.
            2.   MAKE UP A TENTATIVE ORDER.;

3*   REVISE-ORDER
     PROCEDURE;
      FOR EACH BUYER CHECKING OUT:
            1.   IF ORDER FORM MARKED 'NO SALE'
                    CALL CAISSE TO RELEASE BUYER'S FUNDS
                 OTHERWISE
                    COPY ACTUAL AMOUNTS FROM HANGER TICKET ONTO ORDER FORM
                    CALL CAISSE TO CONFIRM ACTUAL AMOUNT OF SALE
                    WRITE SALES RECEIPT FOR BUYER
                    GIVE BUYER YELLOW TICKET COPY OF HANGER TICKET
                    REMAINING TICKETS AND REVISED ORDER COMBINED TO FORM
                          ORDER PACKET.;

4*   SELECT-AND-REGROUP-ITEMS
     PROCEDURE;
      FOR EACH BUYER:
            1.   GUIDE THROUGH SALES AREA AND NEGOTIATE PRICE.
            2.   IF BUYER SELECTS GOODS
                       REMOVE TICKET FROM HANGER
                       MARK CONDITIONS OF SALE ON TOP COPY
                       INITIALIZE BUYER'S ORDER FORM TO SIGNIFY A SALE
                       SEND FILLED-IN-TICKET AND SIGNED ORDER FORM
                             TO MANAGER TO FINALIZE SALE
                 OTHERWISE
                       MARK BUYER'S ORDER FORM 'NO SALE'.
            3.   IF PART OF A HANGER IS SOLD
                       SPLIT HANGER INTO SOLD AND UNSOLD PARTS
                       REVISE THE HANGER TICKET
                       UPDATE THE RECEIPTS FILE.;
```

157

PROCESS SUMMARY

PARAMETERS: DB=CHEESE.DB FILE=PSATEMP.NAME ENTERING LEAVING NOOTHER DESCRIPTION PROCEDURE
PRINT NONEW-PAGE NOINDEX NOPUNCHED-NAMES

1* REJECT-BAD-LOT

FOR EACH BAD LOT:
1. WRITE COMPLAINT ON DELIVERY DOCUMENT.
2. READDRESS LOT BACK TO TRUCK DOCK.
3. SEND BACK VIA THE CONVEYOR.

INFORMATION ENTERING

1 BAD-LOT	ENTITY	USED TO DERIVE

INFORMATION LEAVING

1 REJECTED-LOT	OUTPUT	GENERATED
2 REJECTED-LOT	OUTPUT	DERIVED

2* TICKET-ACCEPTED-LOT

FOR EACH ACCEPTED LOT PLACED IN THE SALES AREA:
1. ACKNOWLEDGE RECEIPT
   1.1 REMOVE DELIVERY DOCUMENT.
   1.2 MARK HANGER NUMBER ON IT.
   1.3 PLACE IT IN THE RECEIPTS FILE.
2. READY LOT FOR SALE
   2.1 MAKE UP A HANGER TICKET.
   2.2 PLACE IN ON THE HANGER.
   2.3 SEND ONE COPY OF THE HANGER TICKET TO REPORTING.

INFORMATION ENTERING

1 DELIVERY-DOCUMENT	ENTITY	USED TO DERIVE
2 HANGER-NUMBER	ELEMENT	USED TO DERIVE

INFORMATION LEAVING

1 ARRIVAL-RECEIPT	ENTITY	DERIVED

3* VERIFY-LOT

FOR EACH DELIVERED-LOT:
1. CHECK FOR DAMAGE OR INCOMPLETENESS.
2. IF LOT IS ACCEPTABLE
      MAKE UP A TRUCKERS RELEASE
      SEND IT VIA PNEUMATIC TUBE
      PLACE ACCEPTED HANGER INTO SALES AREA

FROMAGE-DU-MIDI

PROCESS SUMMARY

```
        OTHERWISE
            LABEL LOT AS BEING BAD.

INFORMATION ENTERING

    1 DELIVERED-LOT             INPUT        RECEIVED
    2 DELIVERED-LOT             INPUT        USED TO DERIVE
    3 DELIVERED-LOT             INPUT        USED TO DERIVE
    4 DELIVERED-LOT             INPUT        USED TO DERIVE

INFORMATION LEAVING

    1 ACCEPTED-LOT              ENTITY       DERIVED
    2 BAD-LOT                   ENTITY       DERIVED
    3 TRUCKERS-RELEASE          OUTPUT       DERIVED
```

DATA ACTIVITY INTERACTION REPORT

THE ROWS ARE DATA NAMES, THE COLUMNS ARE ACTIVITY NAMES.

ROW NAMES

1 SEND-LIST	INPUT
2 IMMED-DELIVERY-TICKET	ENTITY
3 DELIVERY-TICKETS	ENTITY
4 SOLD-ORDER	ENTITY
5 DAILY-REPORTS	OUTPUT
6 SALES-RECEIPT	OUTPUT
7 OUTGOING-SALE	OUTPUT
8 TRUCKERS-RELEASE	OUTPUT
9 REJECTED-LOT	OUTPUT
10 COMPLETED-ORDER	OUTPUT
11 CREDIT-REQUEST	OUTPUT
12 ORDER	INPUT
13 DELIVERED-LOT	INPUT
14 CONFIRMATION	INPUT
15 RETAIL-RECEIPT	INPUT
16 SENT-GOODS-TICKETS	ENTITY
17 ORDER-PACKET	ENTITY
18 RED-TICKET	ENTITY
19 SENT-STAMP	ELEMENT
20 HANGER	ENTITY
21 CLEAN-ORDER-FORM	ENTITY
22 RED-TICKET-COPY	GROUP
23 GREEN-TICKET-COPY	GROUP
24 WHITE-TICKET-COPY	GROUP
25 CLEAN-ORDER-COPY1	GROUP
26 CLEAN-ORDER-COPY2	GROUP
27 REMAINING-TICKET-COPIES	GROUP
28 HANGER-TICKET	ENTITY
29 RECEIPTS-FILE	SET
30 SALES-AREA	SET
31 REVISED-ORDER	ENTITY
32 CHECK-OUT	ENTITY
33 FILLED-II-TICKETS	ENTITY
34 ACCEPTED-LOT	ENTITY
35 BAD-LOT	ENTITY
36 CREDITED-ORDER-FORM	ENTITY
37 ORDER-FORM	ENTITY
38 TICKET-SET	SET
39 TICKET-COPY	ENTITY
40 DAILY-SALES-REPORT	OUTPUT
41 DELIVERY-TIME-REPORT	OUTPUT
42 INVENTORY-STATUS-REPORT	OUTPUT
43 ARRIVAL-RECEIPT	ENTITY
44 DELIVERY-DOCUMENT	ENTITY
45 HANGER-NUMBER	ELEMENT

DATA ACTIVITY INTERACTION REPORT

THE ROWS ARE DATA NAMES, THE COLUMNS ARE ACTIVITY NAMES.

COLUMN NAMES

1	ARRANGE-DELIVERY	PROCESS
2	WHOLESALE-OPERATIONS	PROCESS
3	DISPATCH-GOODS	PROCESS
4	CAISSE-CENTRAL	INTERFACE
5	SEND-HANGER	PROCESS
6	RECORD-ORDER	PROCESS
7	OWNER	INTERFACE
8	GENERATE-REPORTS	PROCESS
9	BUYER	INTERFACE
10	SELLING-GOODS	PROCESS
11	REVISE-ORDER	PROCESS
12	TRUCKER	INTERFACE
13	RECEIVE-LOT	PROCESS
14	VERIFY-LOT	PROCESS
15	REJECT-BAD-LOT	PROCESS
16	CHECK-CREDIT-HOLD-FUNDS	PROCESS
17	DOCUMENT-TENTATIVE-ORDER	PROCESS
18	CREATE-TICKET-SET	PROCESS
19	SELECT-AND-REGROUP-ITEMS	PROCESS
20	SUMMARIZE-SALES	PROCESS
21	REPORT-ON-SERVICE	PROCESS
22	PREPARE-STOCK-REPORT	PROCESS
23	MARK-GOODS	PROCESS
24	TICKET-ACCEPTED-LOT	PROCESS

DATA ACTIVITY INTERACTION REPORT

DATA ACTIVITY INTERACTION MATRIX

(I,J) VALUE	MEANING
R	ROW I IS RECEIVED OR USED BY COLUMN J (INPUT)
U	ROW I IS UPDATED BY COLUMN J
D	ROW I IS DERIVED OR GENERATED BY COLUMN J (OUTPUT)
A	ROW I IS INPUT TO, UPDATED BY, AND OUTPUT OF COLUMN J (ALL)
F	ROW I IS INPUT TO AND OUTPUT OF COLUMN J (FLOW)
1	ROW I IS INPUT TO AND UPDATED BY COLUMN J
2	ROW I IS UPDATED BY AND OUTPUT OF COLUMN J

```
              111111111122222
     123456789012345678901234
    +----+----+----+----+----+
 1  |RKRD|    |    |    |    |
 2  |D   R    |    |    |    |
 3  |D   R    |    |    |    |
 4  |R   |D   |    |    |    |
 5  +-D--+-RD-+----+----+----+
 6  | D  |   RDD   |    |    |
 7  | DDRD    |    |    |    |
 8  | D  |    | RDD|    |    |
 9  | D R|    |  D D    |    |
10  +-DDR+D---+----+----+----+
11  | D R|    DD   |D   |    |
12  | R  |   DR    | R  |    |
13  | R D|    |  RR|    |    |
14  | R D|    R    |R   |    |
15  +-R--+--RD+----+--R-+----+
16  |  D D  R |    |  R |    |
17  |  R |R   DD   |    |    |
18  |    RD   |    |    |    |
19  |    R    |    |    |    |
20  +----R----+----+----+----+
21  |    |F   |    |    |    |
22  |    |F   |    |    |    |
23  |    |F   |    |    |    |
24  |    |F   |    |    |    |
25  |    |F   |    |    |    |
    +----+----+----+----+----+
```

162

FROMAGE-DU-MIDI

DATA ACTIVITY INTERACTION REPORT

DATA ACTIVITY INTERACTION MATRIX

```
              111111111122222
    123456789012345678901234
   +----+----+----+----+----+
26 |    |F   |    |    |    |
27 |    |R   |F   |    |    |
28 |    |  R |  D |  R |    |
29 |    |    U  U |    |    |
30 +----+----U--U-+----+----+
31 |    |    |F   |    |    |
32 |    |    |R   |   D|    |
33 |    |    |R   |   D|    |
34 |    |    |   D|    |    |
35 +----+----+---DR----+----+
36 |    |    |    |D  R|    |
37 |    |    |    |RD  |    |
38 |    |    |    |  D RRR  |
39 |    |    |    |   R|    |
40 +----+----+----+----D----+
41 |    |    |    |    |D   |
42 |    |    |    |    | D  |
43 |    |    |    |    |   D|
44 |    |    |    |    |   R|
45 |    |    |    |    |   R|
   +----+----+----+----+----+
```

DATA ACTIVITY INTERACTION REPORT

```
DATA ACTIVITY INTERACTION MATRIX ANALYSIS

DATA
----

HANGER                      (ENTITY)   (ROW     20) NOT DERIVED BY ANY PROCESS
RECEIPTS-FILE               (SET)      (ROW     29) NOT DERIVED BY ANY PROCESS
RECEIPTS-FILE               (SET)      (ROW     29) UPDATED, BUT NOT USED BY ANY PROCESS
SALES-AREA                  (SET)      (ROW     30) NOT DERIVED BY ANY PROCESS
SALES-AREA                  (SET)      (ROW     30) UPDATED, BUT NOT USED BY ANY PROCESS
ACCEPTED-LOT                (ENTITY)   (ROW     34) DERIVED, BUT NOT USED BY ANY PROCESS
TICKET-COPY                 (ENTITY)   (ROW     39) NOT DERIVED BY ANY PROCESS
ARRIVAL-RECEIPT             (ENTITY)   (ROW     43) DERIVED, BUT NOT USED BY ANY PROCESS
DELIVERY-DOCUMENT           (ENTITY)   (ROW     44) NOT DERIVED BY ANY PROCESS

ACTIVITIES
----------

MARK-GOODS                  (PROCESS)  (COLUMN  23) DOES NOT INTERACT WITH ANY DATA
```

DATA ACTIVITY INTERACTION REPORT

LIST OF ROWS AND COLUMNS FOR THE PROCESS INTERACTION MATRIX

ROW NAMES

1	ARRANGE-DELIVERY	PROCESS
2	DISPATCH-GOODS	PROCESS
3	SEND-HANGER	PROCESS
4	RECORD-ORDER	PROCESS
5	SELLING-GOODS	PROCESS
6	REVISE-ORDER	PROCESS
7	RECEIVE-LOT	PROCESS
8	VERIFY-LOT	PROCESS
9	CHECK-CREDIT-HOLD-FUNDS	PROCESS
10	DOCUMENT-TENTATIVE-ORDER	PROCESS
11	CREATE-TICKET-SET	PROCESS
12	SELECT-AND-REGROUP-ITEMS	PROCESS

COLUMN NAMES

1	ARRANGE-DELIVERY	PROCESS
2	DISPATCH-GOODS	PROCESS
3	SEND-HANGER	PROCESS
4	RECORD-ORDER	PROCESS
5	GENERATE-REPORTS	PROCESS
6	REVISE-ORDER	PROCESS
7	REJECT-BAD-LOT	PROCESS
8	CHECK-CREDIT-HOLD-FUNDS	PROCESS
9	CREATE-TICKET-SET	PROCESS
10	SELECT-AND-REGROUP-ITEMS	PROCESS
11	SUMMARIZE-SALES	PROCESS
12	REPORT-ON-SERVICE	PROCESS
13	PREPARE-STOCK-REPORT	PROCESS

FROMAGE-DU-MIDI

DATA ACTIVITY INTERACTION REPORT

PROCESS INTERACTION MATRIX (INCIDENCE)

THE ROWS AND COLUMNS ARE PROCESS NAMES FROM ABOVE.
AN ASTERISK IN (I,J) MEANS THAT SOMETHING DERIVED
OR UPDATED BY PROCESS I IS USED BY PROCESS J.

```
             1111
    1234567890123
   +----+----+---+
 1 |  * |    |   |
 2 |    *   *|   |
 3 |    *   *|   |
 4 |* **|    |   |
 5 +-*-*+----+---+
 6 | * *|*   |   |
 7 |    *   *|   |
 8 |    | *  |   |
 9 |    |    *   |
10 +----+--*-+---+
11 |    |    |***|
12 |    |*   |   |
   +----+----+---+
```

DATA ACTIVITY INTERACTION REPORT

PROCESS INTERACTION MATRIX ANALYSIS

```
WHOLESALE-OPERATIONS         (PROCESS)                NO INTERACTION, BUT HAS SUBPARTS
GENERATE-REPORTS             (PROCESS)  (COLUMN    5) NO SUCCESSORS FOR THIS PROCESS
SELLING-GOODS                (PROCESS)  (ROW       5) NO PREDECESSORS FOR THIS PROCESS
RECEIVE-LOT                  (PROCESS)  (ROW       7) NO PREDECESSORS FOR THIS PROCESS
VERIFY-LOT                   (PROCESS)  (ROW       8) NO PREDECESSORS FOR THIS PROCESS
REJECT-BAD-LOT               (PROCESS)  (COLUMN    7) NO SUCCESSORS FOR THIS PROCESS
DOCUMENT-TENTATIVE-ORDER     (PROCESS)  (ROW      10) NO PREDECESSORS FOR THIS PROCESS
SUMMARIZE-SALES              (PROCESS)  (COLUMN   11) NO SUCCESSORS FOR THIS PROCESS
REPORT-ON-SERVICE            (PROCESS)  (COLUMN   12) NO SUCCESSORS FOR THIS PROCESS
PREPARE-STOCK-REPORT         (PROCESS)  (COLUMN   13) NO SUCCESSORS FOR THIS PROCESS
MARK-GOODS                   (PROCESS)
                                        NO INTERACTION, BUT IS PART OF ANOTHER PROCESS
TICKET-ACCEPTED-LOT          (PROCESS)
                                        NO INTERACTION, BUT IS PART OF ANOTHER PROCESS
```

FROMAGE-DU-MIDI

NAME SELECTION

```
PARAMETERS:   DB=CHEESE.DB  PRINT  PUNCH=PSATEMP.NAME  EMPTY  SELECTION='PRC AND ATTR=DIAG,D-1'
  ORDER=BYTYPE

   1     CHECK-CREDIT-HOLD-FUNDS             PROCESS
   2     DOCUMENT-TENTATIVE-ORDER            PROCESS
   3     REVISE-ORDER                        PROCESS
   4     SELECT-AND-REGROUP-ITEMS            PROCESS
```

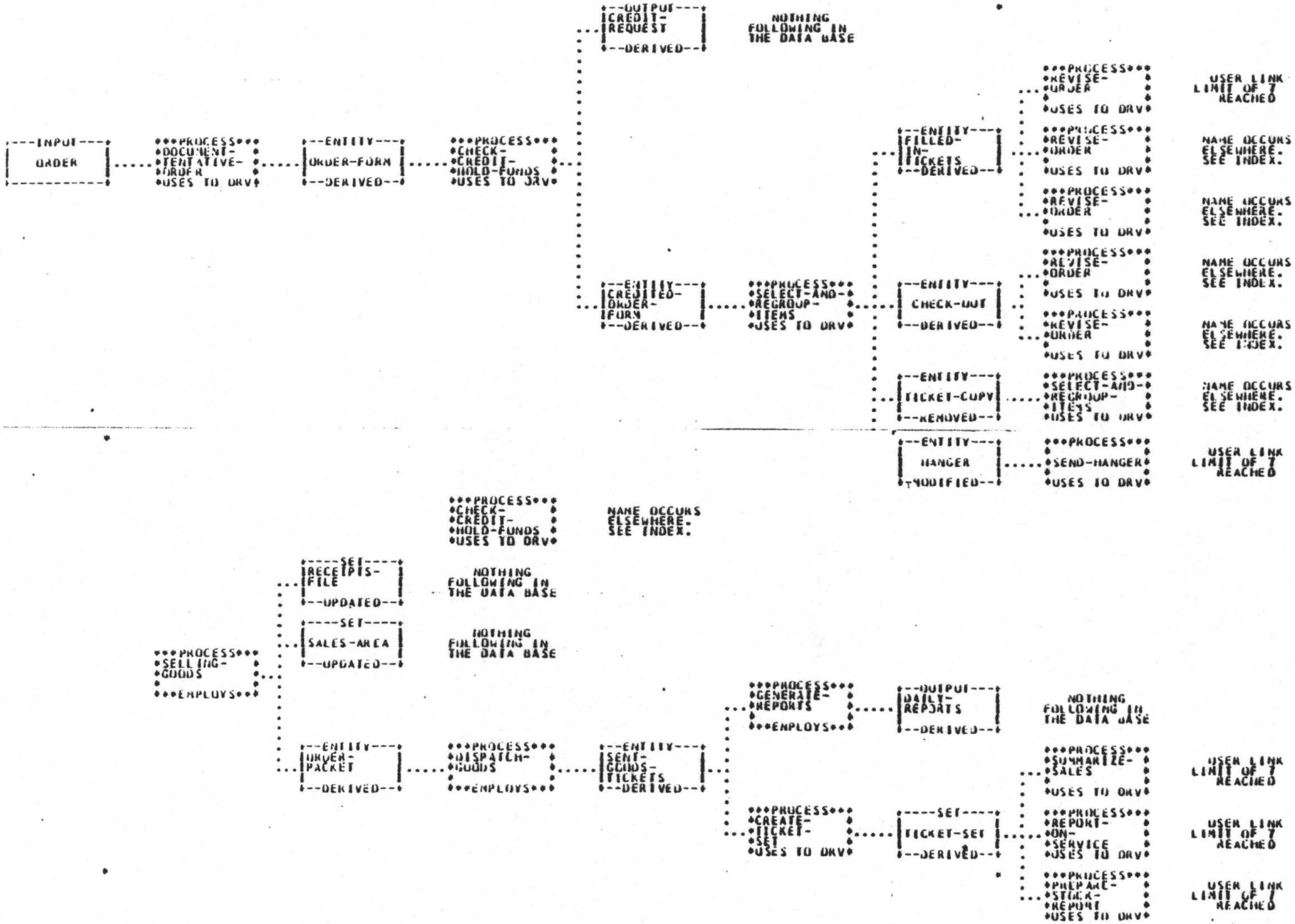
FROMAGE-DU-MIDI

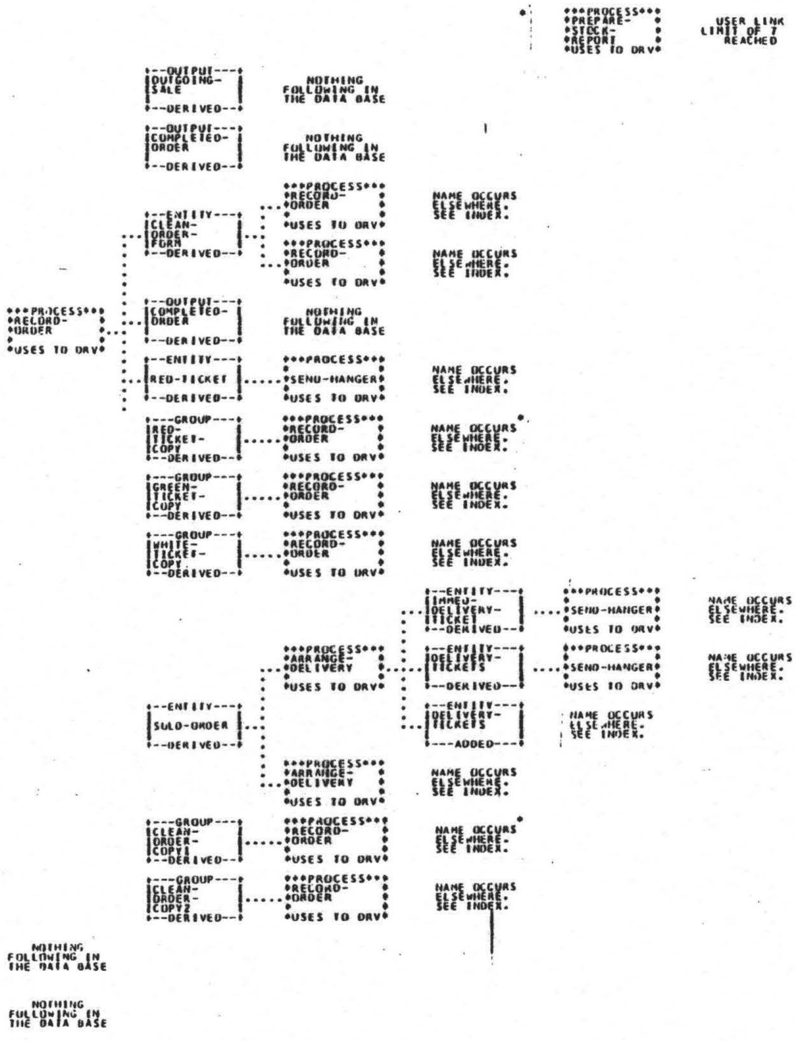
NAME SELECTION

```
PARAMETERS:   DB=CHEESE.DB  PRINT  PUNCH=PSATEMP.NAME  EMPTY  SELECTION='PRC AND NOT SL=2'
  ORDER=BYTYPE

   1      ARRANGE-DELIVERY                  PROCESS
   2      CHECK-CREDIT-HOLD-FUNDS           PROCESS
   3      CREATE-TICKET-SET                 PROCESS
   4      DOCUMENT-TENTATIVE-ORDER          PROCESS
   5      MARK-GOODS                        PROCESS
   6      PREPARE-STOCK-REPORT              PROCESS
   7      RECORD-ORDER                      PROCESS
   8      REJECT-BAD-LOT                    PROCESS
   9      REPORT-ON-SERVICE                 PROCESS
  10      REVISE-ORDER                      PROCESS
  11      SELECT-AND-REGROUP-ITEMS          PROCESS
  12      SEND-HANGER                       PROCESS
  13      SUMMARIZE-SALES                   PROCESS
  14      TICKET-ACCEPTED-LOT               PROCESS
  15      VERIFY-LOT                        PROCESS
```

APPENDIX VIII

EXTENDED PICTURE REPORT


INPUT ORDER
PROCESS DOCUMENT-TENTATIVE-ORDER USES TO DRV
ENTITY ORDER-FORM DERIVED
PROCESS CHECK-CREDIT-HOLD-FUNDS USES TO DRV
OUTPUT CREDIT-REQUEST DERIVED
NOTHING FOLLOWING IN THE DATA BASE
ENTITY CREDITED-ORDER-FORM DERIVED
PROCESS SELECT-AND-REGROUP-ITEMS USES TO DRV
ENTITY FILLED-IN-TICKETS DERIVED
PROCESS REVISE-ORDER USES TO DRV
USER LINK LIMIT OF 7 REACHED
PROCESS REVISE-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS REVISE-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
ENTITY CHECK-OUT DERIVED
PROCESS REVISE-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS REVISE-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
ENTITY TICKET-COPY REMOVED
PROCESS SELECT-AND-REGROUP-ITEMS USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
ENTITY HANGER MODIFIED
PROCESS SEND-HANGER USES TO DRV
USER LINK LIMIT OF 7 REACHED
PROCESS CHECK-CREDIT-HOLD-FUNDS USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
SET RECEIPTS-FILE UPDATED
NOTHING FOLLOWING IN THE DATA BASE
SET SALES-AREA UPDATED
NOTHING FOLLOWING IN THE DATA BASE
PROCESS SELLING-GOODS EMPLOYS
ENTITY ORDER-PACKET DERIVED
PROCESS DISPATCH-GOODS EMPLOYS
ENTITY SENT-GOODS-TICKETS DERIVED
PROCESS GENERATE-REPORTS EMPLOYS
OUTPUT DAILY-REPORTS DERIVED
NOTHING FOLLOWING IN THE DATA BASE
PROCESS CREATE-TICKET-SET USES TO DRV
SET TICKET-SET DERIVED
PROCESS SUMMARIZE-SALES USES TO DRV
USER LINK LIMIT OF 7 REACHED
PROCESS REPORT-ON-SERVICE USES TO DRV
USER LINK LIMIT OF 7 REACHED
PROCESS PREPARE-STOCK-REPORT USES TO DRV
USER LINK LIMIT OF 7 REACHED


PROCESS PREPARE-STOCK-REPORT USES TO DRV
USER LINK LIMIT OF 7 REACHED
OUTPUT OUTGOING-SALE DERIVED
NOTHING FOLLOWING IN THE DATA BASE
OUTPUT COMPLETED-ORDER DERIVED
NOTHING FOLLOWING IN THE DATA BASE
PROCESS RECORD-ORDER USES TO DRV
ENTITY CLEAN-ORDER-FORM DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS RECORD-ORDER USES TO DRV
OUTPUT COMPLETED-ORDER DERIVED
NOTHING FOLLOWING IN THE DATA BASE
ENTITY RED-TICKET DERIVED
PROCESS SEND-HANGER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
GROUP RED-TICKET-COPY DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
GROUP GREEN-TICKET-COPY DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
GROUP WHITE-TICKET-COPY DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
ENTITY IMMED-DELIVERY-TICKET DERIVED
PROCESS SEND-HANGER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS ARRANGE-DELIVERY USES TO DRV
ENTITY DELIVERY-TICKETS DERIVED
PROCESS SEND-HANGER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
ENTITY SOLD-ORDER DERIVED
ENTITY DELIVERY-TICKETS ADDED
NAME OCCURS ELSEWHERE. SEE INDEX.
PROCESS ARRANGE-DELIVERY USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
GROUP CLEAN-ORDER-COPY1 DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
GROUP CLEAN-ORDER-COPY2 DERIVED
PROCESS RECORD-ORDER USES TO DRV
NAME OCCURS ELSEWHERE. SEE INDEX.
OUTPUT CREDIT-REQUEST DERIVED
NOTHING FOLLOWING IN THE DATA BASE
OUTPUT SALES-RECEIPT DERIVED
NOTHING FOLLOWING IN THE DATA BASE

REFERENCES

[ACK71] Ackoff, Russel L. (1971): "Towards a System of Systems Concepts", in [BEI72], pp. 105-112.

[ALT80] Alter, Steven, DECISION SUPPORT SYSTEMS: Current Practices and Continuing Challenges, Addison-Wesley Publishing Company, Don Mills, Ontario, 1980.

[BAR78] Barrow, Howard S., Feightner, J.W., Neufeld, V.R., and Norman, G.R., "An Analysis of the Clinical Methods of Medical Students and Phsicians", A Report submitted to The Province of Ontario Department of Health, and Physician Services Inc. Foundation, McMaster University, March, 1978.

[BEI72] Beishon, J. and Peters, Goeff (eds.), Systems Behaviour, Harper and Row, New York, 1976 (2nd ed.).

[BER72] Bertalanaffy, Ludwig von (1972): "The History and Status of General Systems Theory", in: [COU74], pp. 9-26.

[BUR80] Burnstine, D.C., "BIAIT - An Emerging Management Engineering Discipline", Fox Hollow, Petersburg, New York, 1980.

[CAR79] Carlson, Walter M. (1979), "Business Information Analysis and Integration Technique (BIAIT) - The New Horizon", in: Data Base, Volume 10, Number 4, Spring 1979. pp. 3-9.

[CHE72] Checkland, P.B. (1972): "Towards a Systems-Based Methodology For Real-World Problem Solving", in: [BEI72], pp. 51-77.

[CHE80] Chen, Peter P.(ed.), Entity-Relationship Approach to Systems Analysis and Design, North-Holland Publishing Company, Amsterdam, 1980.

[COU73] Cougar, J.D. (1973): "Evolution of Business Systems Analysis Techniques", in: [COU74], pp. 43-82.

[COU74] Cougar, J.D. and Knapp, R.W.(eds.), Systems Analysis Techniques, John Wiley and Sons, New York, 1974.

[DEM78] DeMarco, Tom, Structured Analysis and Systems Specification, Englewood Cliffs, N.J. Prentice Hall, Inc., 1978.

[EME69] Emery, F.E.(ed.), Systems Thinking, Penquin Books, Suffolk, Great Britain, 1969.

[GAN79] Gane, C. and Sarson, T., Structured Systems Analysis: Tools and Techniques, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1979.

[GOO81] Goossen, J.W. and Welke, R.J., "Systems Mapping of PSL/PSA to Structured Analysis (With an Application to a YOURDON Case Study)", ISRAM Discussion Paper, DP-8101-2.0, McMaster University, Hamilton, Ontario, January 1981.

[ISD79] ISDOS Project, Problem Statement Analyzer Report and Report Commands, August 1979, 79A51-173-3.

[ISD79] ISDOS Project, Problem Statement Language Introduction and User Manual, September 1979, 79A51-0171-1.

[KAS70] Kast, F.E. and Rosenzweig, J.E. (1970): "The Modern View: A Systems Approach", in: [BEI72], pp. 15-29.

[KAT66] Katz, D. and Kahn, R.H. (1966): "Properties of Open Systems", in: [EME69], pp. 86-104.

[KEE78] Keen, Peter G.W. and Scott Morton, Michael S., DECISION SUPPORT SYSTEMS: An Organizational Perspective, Addison-Wesley Publishing Company, Don Mills, Ontario, 1978.

[KER79a] Kerner, David V. (1979), "Business Information Characterization Study", Data Base, Volume 10, Number 4, Spring 1979. pp. 10-17.

[KER80] Kerner, David V., "Introduction to Business Information Control Study Methodology (BICS)", Technical Report, TR 03.113, Santa Teresa Laboratory, San Jose, California, September 1980.

[KER79b] Kerola, Pentti, "On Infological Research into the Systemeering Process", Institute of Data Processing Science, University of Oulu, Finland, 1979.

[KLE80] Klein, H.K. and Welke, R.J., "An Evaluation of the Finnish PSC Systemeering (Systems Development) Theory", ISRAM Position Paper PP-7912-1.1, McMaster University, Hamilton, Ontario, July 1980.

[KOB73] Koberg, Don and Bagnall, Jim, The Universal Traveler, William Kaufmann, Inc., Los Altos, California, 1972.

[LIE80] Lieberman, Arthur Z. (1980), "Mechanized Analysis of Entity-Relationship Design Databases", in: [CHE80], pp. 641-663.

[LIE77] Lieberman, Arthur Z., The ERA/ERE System User Manual, 1977.

[LUC76] Lucas, Henry C., The Analysis, Design and Implementation of Information Systems, McGraw-Hill Book Company, New York, 1976, p.46.

[MAT78] Mattessich, Richard, Instrumental Reasoning and Systems Methodology, D.Reidel Publishing Company, Netherlands, 1978.

[MEN80] Mendes, Kathleen S. (1980), "Structured Systems Analysis: A Technique to Define Business Requirements", in: Sloan Management Review, Volume 21, Number 4, Summer 1980.

[MUM79] Mumford, Enid, and Weir, Mary, Computer Systems in Work Design - the ETHICS Method, John Wiley & Sons, Toronto, 1979. p. 29.

[ROS73] Rosen, B.K. (1973), "Tree-Manipulating Systems and Church-Rosser Theorems", in: Journals of the ACM, Volume 20, No. 1, January, 1973. pp.160-187.

[ROS76] Ross, D.T. (1976), "Structured Analysis (SA): A Language for Communicating Ideas", in: IEEE Transactions on Software Engineering, Volume SE-3, January, 1977.

[TEI76] Teichroew, D. and Hershey, E.A. (1976), "PSL/PSA: A Computer-Aided Technique for Stuctured Documentation and Analysis of Information Processing Systems", in: IEEE Transactions on Software Engineering, Volume SE-3, January, 1977.

[TRE80] Tremblay, J.P., Sorenson, P.G., Wig, E.D. and Perkins, D.R., "A Survey of Automated Aids For System Analysis and Documentation" Department of Computer Science, University of Saskatoon, Saskatchewan, February, 1980.

[WEL77] Welke, R.J. (1977), "Current Information System Analysis and Design Approaches: Framework, Overview, Comments and Conclusion For Large-Complex Information System Education", in: Education and Large Infromation Systems, Buckingham, R.A. (ed.), North-Holland Publishing Company, Amsterdam, 1977. pp. 149-166.

[WEL80a] Welke, R.J., Goossen, J.W. and Kumar, K., "BIAIT BICMX Analysis Using PSL/PSA", ISRAM Working Paper, WP-8009-1.1, McMaster University, Hamilton, Ontario, September 1980.

[WEL80b] Welke, R.J. and Kumar, K., "An "ERA"-Based Analyst Support System for BIAIT", ISRAM Working Paper, WP-8010-2.0, McMaster University, Hamilton, Ontario, October 1980.

[WIG79] Wig, Eldon D., "Computer-Aided Systems Analysis Reporting", Department of Computer Science, University of Saskatoon, Saskatchewan, June, 1979.

[WIN79] Winters, Edward W., "An Analysis of the Capabilities of Problem Statement Language: A Language for System Requirements and Specifications, Proceedings COMPSAC 79, IEEE Third International Computer Software & Applications Conference, November 5-8, 1979, Chicago, IL. pp. 283-288.

[YOU79] YOURDON inc, Structured Analysis and System Specification Workbook, 5th ed., 1979.