Industry 4.0 Digital Foundation

A LEARNING FACTORY CASE STUDY: INDUSTRY 4.0 DIGITAL FOUNDATION

By MOHAMED ALY, B.SC.

A Thesis submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Master of Applied Science

McMaster University MASTER OF APPLIED SCIENCE (2018) Hamilton, Ontario

TITLE: A Learning Factory Case Study: Industry 4.0 Digital Foundation AUTHOR: Mohamed Aly SUPERVISOR: Professor Mohamed Elbestawi NUMBER OF PAGES: viii, 154

Acknowledgments

First of all, I would like to thank God for giving me the strength to finish the below work. Next, I would like to thank my supervisor Dr. Elbestawi for his support to me throughout this transitional phase in my life. I would also like to thank my entire family and relatives who stood by me in such time of need specially my wife Sarah and kids Yousef and Joumana who endured me through those two years in our a new country.

Abstract

The fourth industrial revolution is the extension of the third digitization revolution in terms of integrating computers and software to transform manufacturing to become green, customizable, on demand, and as a service. In the work done below, we elaborate on the information and communication technology architecture enabling this revolutionary transformation. We show the way to move forward with technology from as low level as sensors and all the way up till we reach services and customization. All this work is done under the roof of a learning factory establishment, which yields the way for not only knowledge transfer and hands-on experience, but also research and development in collaboration with market-leading software, hardware, communication, and manufacturing companies.

Table of Contents

Table of Figures

CHAPTER 1 INTRODUCTION

The Fourth Industrial Revolution is here and happening now, whether it is called Industry 4.0 as per Germany and majority of European countries establishments or Smart Manufacturing as per North American manufacturing societies. Nearly all devices produced by manufacturers are internet connectivity enabled and already or ready to be registered on any network since production. In addition, any device history is being logged since it was just a design on a software in the cloud and similarly all parts are being tracked from design to manufacturing to installation in a bigger system. Everything is being recorded and logged digitally, and ready to be used in analytics to generate a new feedback loop aimed to enhance not only the products quality, customer experience and performance, but also manufacturers and consumers way of doing business and life. New revenue streams, customer tailored products, optimized and customized production, smart services and prevention technologies are all made possible by Industry 4.0. If we are to take car drivers as an example, they do not need to worry anymore about car service, parts failure, accidents, etc. since everything now is available as a service few clicks away from subscribers where companies are moving towards subscription economy backed by Cloud services, e.g., Warranty as a Service, Maintenance as a Service and Emergency/Roadside-Assistance as a Service. Everything is customer focused allowing for more freedom of choice and newly offered unmatched customization options made possible by Industry 4.0 customized production and cloud enabled smart services.

The Industrial revolution changes originated from the manufacturing evolution starting with pure handmade items, to mechanical/machines-assisted production, to computer

operated, and eventually, to a fully automated remote supervised manufacturing. This remarkable evolution is now moving past mechanical and computer integration and automation into a data driven revolution, where all manufacturers log their data generated from every design, process, implementation, production, usage, customer interaction, and much more using the Internet of Things platform. Consequently, the computers function in industry shifted from only data processing to encompass enormous data storage and advanced data analytics. In other words, the exponentially growing Big Data generated from the Internet of Things needs an exponentially growing storage facilities to store, process and backup. Also, growing manufacturing processing and complex designs require significant computer resources in terms of compute and network. Thus, the need for more storage, network, and compute to maintain the overwhelming growth, which is made possible with the introduction of the cloud computing, where computing resources is available as on-demand self-service with broad network access, resource pooling, rapid elasticity, measured services and pay-as-you-go model. Combining Cloud Computing with Internet of Things, we have the basics for the fourth industrial revolution. Accordingly, the Industry 4.0 emerging terms, e.g., Cloud Manufacturing (CMfg), Industrial Internet of Things (IIoT), Cyber-Physical Production Systems (CPPS), Manufacturing as a Service (MfgaaS), etc.

In this Industry 4.0 era overwhelmed by Information and Communication Technology (ICT), university graduates face tremendous pressure to certify and get hands-on trainings before qualifying to work with companies. A university degree without any hands-on training is not enough anymore, which marked the noticeable change in many universities to integrate both theoretical and practical education as a part of their curriculum under what is commonly known as a Learning Factory (LF). LFs are

becoming more and more integrated into education from early semesters at many universities, where students are showing growing interest in hands-on projects focused on training them on current technologies, and designing and implementing research for future technologies funded by partners from education, government and industry.

In this research work, we introduce the ICT bottom up infrastructure digital foundation transforming LFs to become Industry 4.0 Smart Manufacturing ready. This transformation is possible by reusing existing IT infrastructure and applying virtualization technology on top of it to fully utilize its distributed resources into one virtual pool. In addition, we apply clustering technologies to ensure high availability and redundancy. Once infrastructure is virtualization and cluster ready, we use cloud management software tools to create service models/levels transforming the environment into a local private cloud. Subsequently, we integrate the cyber-physical systems (sensors, actuators, etc.) of the LF manufacturing assembly line(s) stations with our private cloud for monitoring, reporting, management and advanced digital control. Consequently, to be able to leverage the industry 4.0 capabilities, we link our private cloud to the public cloud creating a hybrid cloud model on which we utilize the pool of available resources to safe keep all historical data/transactions allowing us to apply the big data analytics software. Moreover, by creating a hybrid cloud we facilitate our production services to be available for customization by connected customers who in turn function as an additional source of data under the big umbrella of Internet of Things. By providing subscription, customization, and many more services to the customer through service oriented paradigm, e.g., distributed computing applications, integrated modules, and services, we are fully utilizing Industry 4.0 ICT infrastructure technologies. Finally, we dive into Green Industry 4.0 technologies and suggest best

practices based on the literature review and market hands-on experience, during which, we introduce a modified Internet of Things (IoT) algorithm offering lower complexity and execution time following the footsteps of an IEEE IoT Journal published paper algorithm.

The novelty of this work lies in the Industry 4.0 digital foundation architectural design aimed to provide an ecosystem upon which applications and services can run providing customization and subscription services to public. These subscription and customization services are the core of today's world, for example, each car buyer wants to customize and tailor his car upon his specific needs, in the country of his desire and with the options that would best fit him. Therefore, the need for manufacturers' to provide the tools enabling the end user to customize their products; thus, the fourth industrial revolution main goal, which is enabling users not only corporations. Subsequently using Industry 4.0 digital foundation architectural design, we optimize existing manufacturing sensing nodes to use a less complex chipsets as we move the feedback loop to the private cloud, minimizing the computation needed on the sensing nodes; accordingly, improving the overall system budget, data transfer and energy consumption on each sensing node. Using the above design, and optimization functions and constraints we re-solve an NP-Hard problem allowing sensing nodes in manufacturing production environment to be able to use less energy, have faster and more efficient data transfer, and reduced overall cost/budget.

CHAPTER 2 LITERATURE REVIEW

Learning Factory

The learning factory concept originally known as “Lernfabrik” is a purely German idea that started in 1988. It was established with the goal of delivering education and training using new techniques in production to graduate qualified interdisciplinary educated employees. It was equipped with top class Computer-Aided Design and Manufacturing (CAD/CAM) software system workstations for planning and design education and training. In addition, the “Lernfabrik” emphasized hands-on production planning and control through Computer Numerical Control (CNC) production machine tools (Reith). The learning factory goal was to graduate industry-ready calibers ready for integration in small and midsize enterprises.

The term “Learning Factory” was first introduced in the United States of America (USA) in 1994 by the National Science Foundation (NSF), Advanced Research Projects Agency (ARPA), later known as Defense ARPA (DARPA), and Manufacturing Engineering Education Partnership (MEEP). The learning Factory was one of many collaborative initiatives between NSF and DARPA in various interdisciplinary fields to support and advocate research and education without boundaries. NSF and DARPA funded MEEP with a three-year technology reinvestment program grant in manufacturing engineering education to create the first learning factory in the USA. MEEP is a partnership between Penn State University (PSU), the University-of-Washington (UW), the University of Puerto Rico-Mayaguez (UPRM), Sandia National Laboratories, and twenty-four other corporate partners. This partnership initiated the learning factory concept in the USA promoting practice-based educational facilities

offering customized manufacturing tailored curriculum developed by the participating universities of the MEEP (Lamancusa et al.; Jorgensen et al.).

Throughout the years, manufacturing evolved in many ways integrating new toolsets, hardware and software technologies, and diverse various advancements in modeling, simulation, automation, customization, and production. This marks the urgent need for manufacturing education to build capacities capable of coping with the fast-paced technological advancements. Thus, continuous learning and hands-on training became a must in this era to maintain a knowledgeable capable set of workers and engineers (Jovane et al.). Early adoption of hands-on training in a learning factory environment proved to reshape the future of students allowing them to be more productive, efficient and successful (Cachay et al.).

Manufacturing skilled, competent and market ready students are in high demand; however, the supplies from fresh only theoretically educated students are not meeting the manufacturers' minimum requirements and expectations. As a result, many initiatives have been taken by universities and governments to research and build learning factories to bridge the gap between university graduates and industrial manufacturing needs. Consequently, manufacturing education has been introduced in universities through the learning factory to produce skilled and competent graduates with associate level industrial practice on most common manufacturing techniques (Chryssolouris et al.; Rentzos et al.).

According to the first and only review paper on the learning factories found in the literature, the learning factories are classified into eleven categories as follows (Abele et al.):

- LF(s) for Production Process Improvement.
- LF(s) for Re-configurability, Production and Layout Planning.
- LF(s) for Energy and Resource Efficiency.
- Applied Teaching Factory Concept.
- LF(s) for Industry 4.0.
- LF(s) for Sustainability.
- LF(s) for Product Emergence Process.
- LF(s) for Logistics Optimization.
- LF(s) for Management and Organization.
- LF(s) for Business Administration.
- LF(s) for Automation Technology.

In our research below, the focus is only on the learning factories for Industry 4.0 found in less than ten universities worldwide, which is a small number given the fourth industrial revolution estimated market value and impact, and also, the companies tendencies to immediately hire students affiliated with modern state of the art learning curricula integrated with hands-on experience and up-to-date research and development.

Below, we dive into Industry 4.0 learning factory universities starting with Vienna University of Technology (TU Wein), which is undergoing the establishment of the first Industry 4.0 Learning Factory in Austria, which they named Industry 4.0 Pilot Factory (I40PF). The I40PF focuses on future production research, teaching, and training, where it extends the well-established teaching and training cross-disciplinary and practical education former platform known as Learning and Innovation Factory for Integrative

Production Education (LIF). LIF is based on Integrative Product Engineering Process (I-PEP) enabling students' hands-on production process learning. In other words, the I40PF extends LIF and I-PEP concepts to the future Industry 4.0 competencies, while also extending the learning groups from only students to managers, engineers, and workers from industrial firms [9].

I40PF has five main objectives, which start with smart production research, where the pilot factory in cooperation with partners from ICT develop new concepts, models, technologies, and systems. Second, I40PF provide access to technology, research, development, and testing for companies with no research infrastructure. Third, I40PF provides contract manufacturing for companies, especially startups, looking to produce prototypes and small lot series. Fourth, research results in knowledge transfer to students to highly educate them on the innovative future production infrastructure. Finally, I40PF offers advanced education, especially in digitization, through hands-on workshops and seminars (Erol et al.).

To build the I40PF, TU Wein partnered with twenty industrial technology solutions suppliers to establish the state-of-the-art factory focused on Industry 4.0. They focus on Cyber-Physical Production Systems (CPPS) implemented through three main systems, i.e., cyber-physical manufacturing system, cyber-physical assembly system, and cyber-physical adaptive logistics system. In the figure below, TU Wein shows us an example of one of the CPPS, i.e., the future cyber-physical assembly system made of a collection of connected mobile assembly stations (Erol et al.).

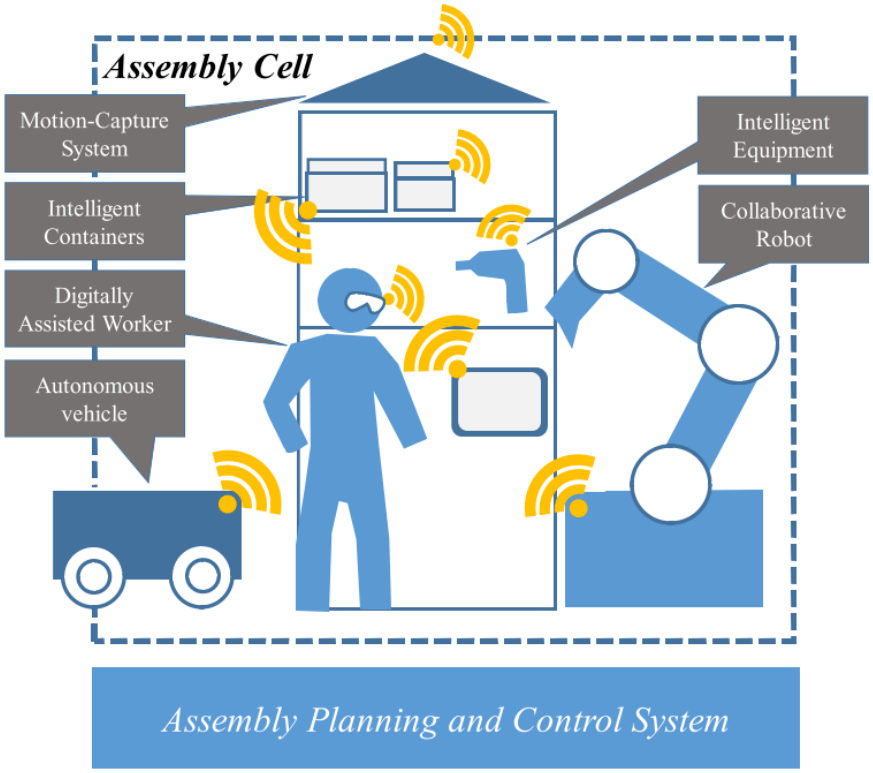

Figure 1 - Assembly cell concept for the TU Wien Pilot Factory (Erol et al.)

Paderborn University (PU) Heinz Nixdorf Institute (HNI) is in the process of developing two learning Factories focused on Product Creation Process (PCP) and the Cyber-Physical Production Systems (CPPS) (Gräßler, Taplick, et al.; Gräßler, Pöhler, et al.). First, the PCP Learning Factory is based on a Product Lifecycle (PLC) approach with more focus and interaction between education and research. PCP deploys various complex product development phases including Industry 4.0 future development concepts based on mechanics, electronics, and informatics; in addition, PCP relies on innovation management, engineering methodologies and management, integrated production management and virtual engineering (Gräßler, Taplick, et al.). In the concept image below, it shows the focus on the holistic approach of the learning factory on the whole lifecycle of a product.

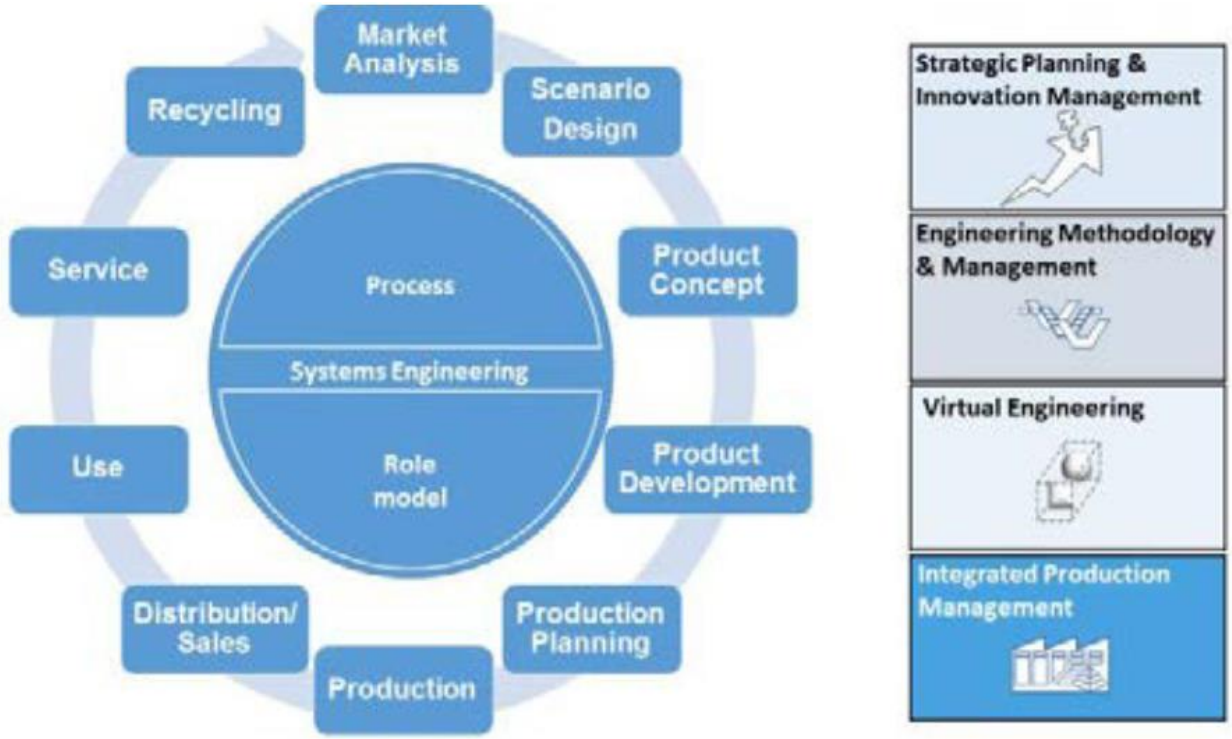

Figure 2 - PU HNI PCP Learning Factory concept (Gräßler, Taplick, et al.)

The second PU HNI Learning Factory (LF) is the CPPS LF, which is an upgrade to an existing laboratory for centralized production, which is focused on production and automation engineering. CPPS LF implements CPPS on the basis of single-board computers (SBC) containing communication devices, and information gathering, and processing tools connected to each unit of production cyber-physical devices. The CPPS LF focuses on six main pillars, which are flexible reconfiguration of production control systems, decentralization of decision-making and execution, modularization of the production system, adaptive connection of all production participants, plug and Produce, and, lastly, decentralized production planning system (Gräßler, Pöhler, et al.). The PU NHI CPPS LF is implemented on the basis of Single Board Computer (SBC) replacing the former PLC-based control system, where each production systems unit is

equipped with an SBC transforming it into a CPS capable of gathering, communicating, and processing information as per the image below of the LF.

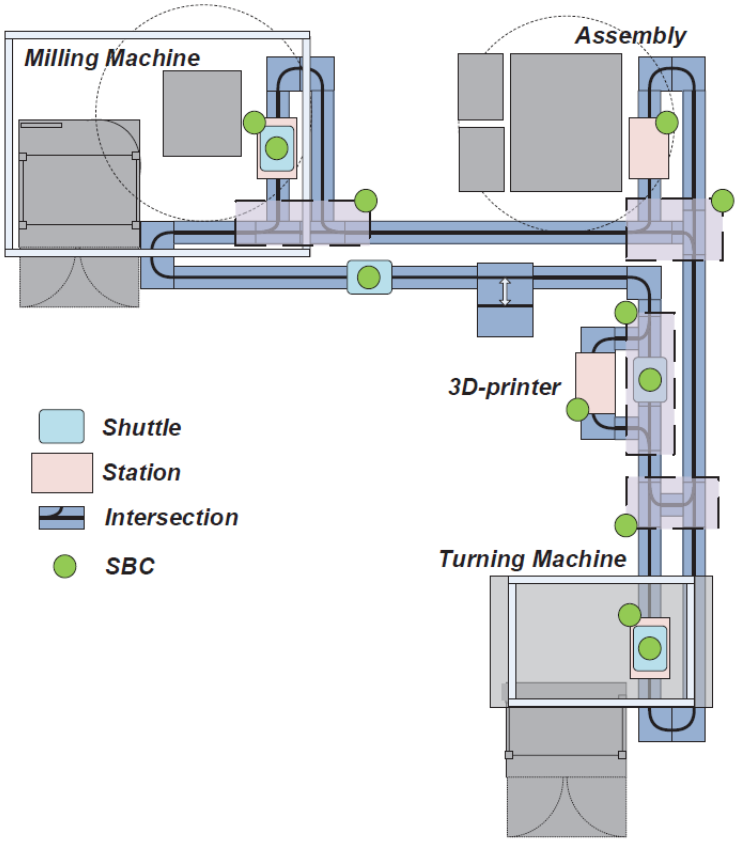

Figure 3 – PU HNI CPPS LF layout with SBC (Gräßler, Pöhler, et al.)

Ruhr University Bochum (RUB) Lehrstuhl für Produktionssysteme (LPS), i.e., Chair of Production Systems, Learning Factory is equipped like small and mid-size enterprise (SME), where they have two fully equipped assembly lines containing seven stations per line. The LPS LF has the goal of simulating real production system use cases with real-world products. The first assembly line is focused on only a single production process improvement and industry 4.0 new technologies implementation, while the second assembly line focus is on new ideas implementation on any product and technology of choice (Prinz et al.). On another streamline, RUB is planning to expand

their Learning Factory to become Learning Company simulating other schools offering, for example, simulated court sessions and business stock market sessions. In additions, as a part of RUB future plans, the university aims to lunch two huge projects in partnership with several universities, commercial enterprises and partnered companies; the "PTR" (Produktionstechnisches Trainings- und Forschungszentrum Ruhr) is to be a center for education and research, while the World Factory to be a host of multidisciplinary real-world projects with partner companies (Bender et al.).

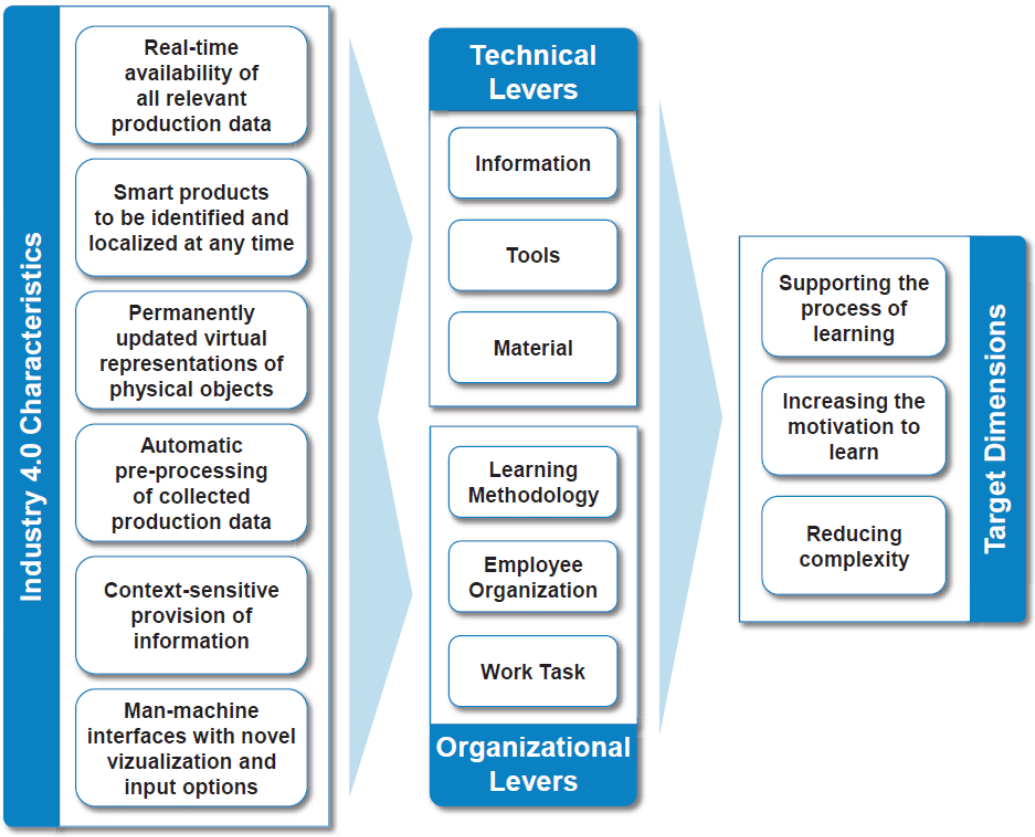

Figure 4 - RWTH Aachen LF work-based learning (Schuh, Gartzen, et al.)

RWTH (Rheinisch-Westfälische Technische Hochschule) Aachen University contains a Learning Factory, which they call "Demonstration Factory", where they uphold hybrid production and Industry 4.0 work-based learning (Schuh, Gartzen, et al.; Schuh, Prote,

et al.). They use a combination of research and production called hybrid production, which improves industrial learning. The Demonstration Factory transforms experimentation learning into the real-time productive manufacturing process of actual world problems (Schuh, Prote, et al.). In addition, as per image below, the RWTH LF work on modeling learning curve from individual employee to the entire production line, where they use scientific theory and real production environment combined with structured evaluation approach to reach effective evaluations and results of work-based learning (Schuh, Gartzen, et al.).

Leibniz Universität Hannover (LUH) has a well-established Learning Factory called Institut für Fabrikanlagen und Logistik (IFA), i.e., Institute of Production Systems and Logistics, which uses digital learning games as one of its ways of knowledge transfer to students. The real edge in digital learning games is the ability to reproduce problems and solutions autonomously from spatial and temporal framework conditions. In addition, access to these games are not solely limited to the learning factory floor, but also it could be reused after to recreate the same solutions and learning environment (Görke et al.).

The IFA LF focuses on designing and controlling production systems, specifically on the organization of manufacturing and assembly processes. They cover a variety of topics, for example, lean production, production monitoring, ergonomics, factory planning and production planning control (PPC). They designed a simple IT infrastructure model and integrated it with PPC to facilitate implementing logistics models on top of near-real-time captured data, which improves production monitoring and planning control. In addition, through their model, they show the advantages of

cyber-physical-production-systems (CPPS) in planning, monitoring and controlling production systems (Seitz and Nyhuis).

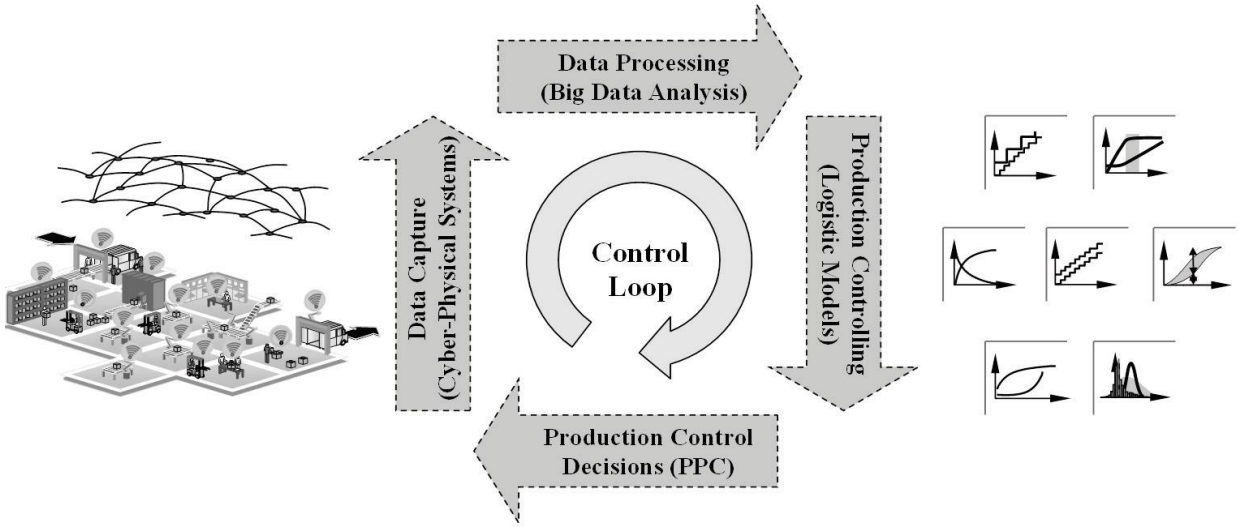

Figure 5 - LUH IFA LF CPPS with Logistics Model (Seitz and Nyhuis)

Technische Universität Braunschweig (TUB) Institute of Machine Tools and Production Technology (IWF) contains a learning factory focused on Industry 4.0 rapid development of production engineering and information communication technology, which is realized through the implementation of cyber-physical production systems (Thiede et al.). TUB IWF LF focuses on CPPS as it presents a huge part of the future of manufacturing, education, and training, which is essential to the success and existence of all companies the new era of Industry 4.0. A student action-based learning project was conducted not only educating CPPS but also implementing it and testing it, which yielded great results and popularity among students. Students were able to focus on modeling, implementation and testing breaking the loop of only being knowledge recipients, but now also hands-on applicants of technological advancements and contributors to the fourth industrial revolution (Thiede et al.).

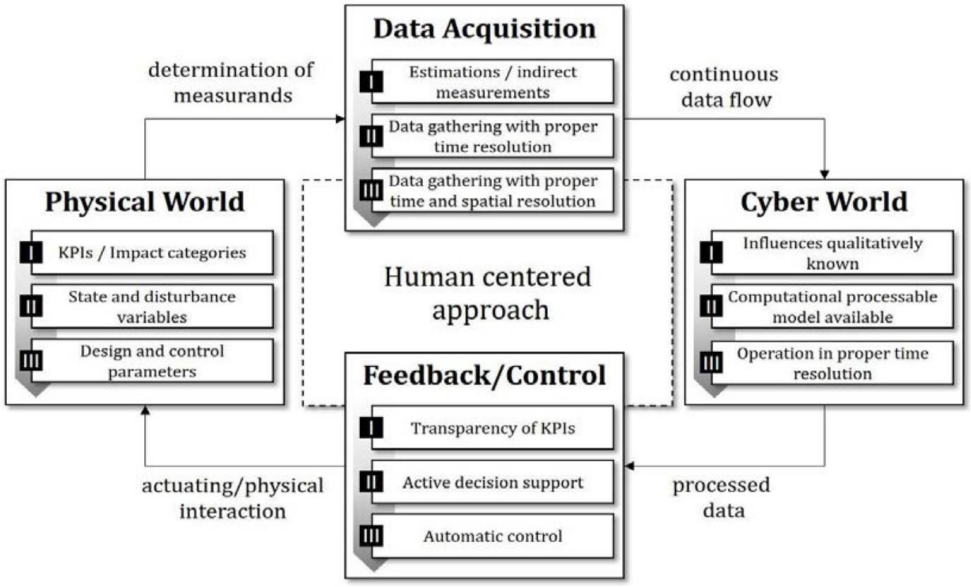

Figure 6 – TUB IWF LF CPPS structure assessment procedure analysis and development (Thiede et al.)

Additionally, TUB IWF LF works on closed-loop additive manufacturing, where they take end-of-life product materials and components through a recycling process of crushing and extruding new filament out of shredded material, which is later utilized to create new products (Juraschek et al.). This newly implemented closed-loop process chain with continuously recycled material utilization provides a rich environment for learning, practical experience, and research in engineering closed-loop production systems education. Students engaged in learning, developing and implementing the system not only acquiring hands-on experience, but also many soft skills required in the workforce, e.g. communication skills.

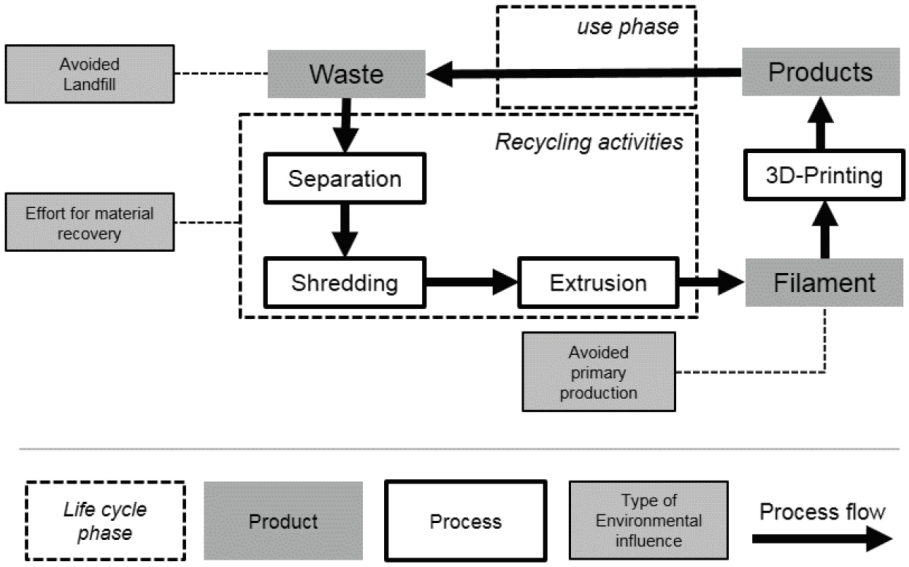

Figure 7 - TUB IWF LF closed-loop additive manufacturing production process steps (Juraschek et al.)

Technische Universität Darmstadt (TUD) already have an existing production learning factory on which they launched a research project to implement Industry 4.0 technologies (Wank et al.). In the existing TUD LF, a well-established IT-infrastructure exists, which they shall utilize and reconfigure to match Industry 4.0 technologies increasing its efficiency and flexibility. The main objective of the research project and competence center is to build capacities of local subject matter experts, who are able to digitize, interconnect and implement Industry 4.0 technologies using existing capable smart objects and communication techniques to create a more efficient and valuable production process chain, and generate new market opportunities through increased customer engagement (Wank et al.).

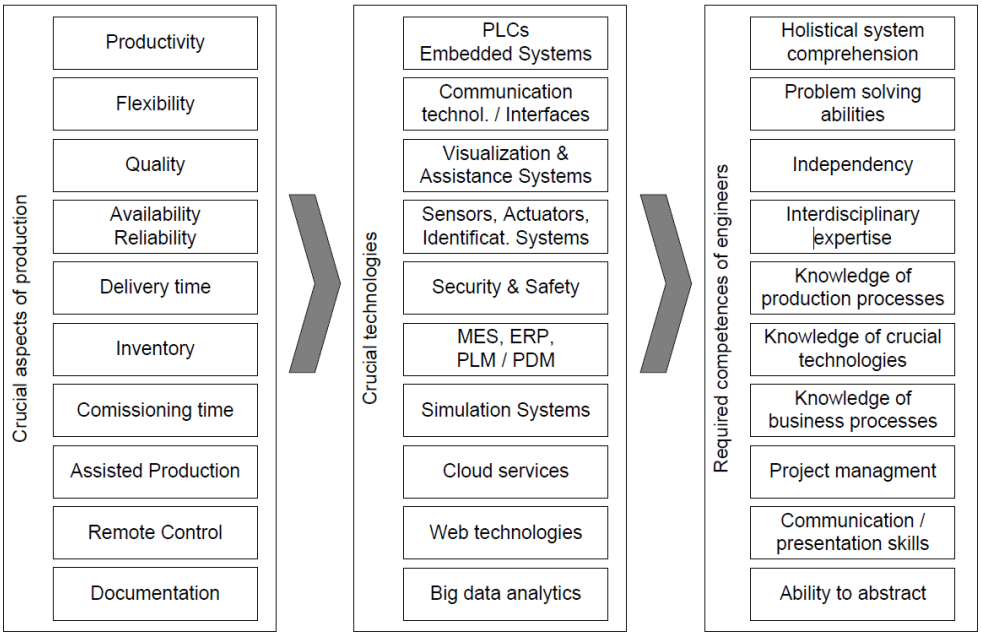

Figure 8 - H_Da LF - Production, Technologies and Competencies (Simons et al.)

The Darmstadt University of Applied Sciences, also known as Hochschule Darmstadt (H_Da), established a fully automated Industry 4.0 learning factory, named AutFab, to educate students on Industry 4.0 technologies to meet the changing manufacturing industry demand shifting from mass production to customized production, where the students' engagement is either in semester project-based courses or problem-based lab work (Simons et al.). Students' engagement in the LF has proven to be successful in linking theoretical and practical knowledge to the extent that they are requesting to have earlier curriculum engagement in the AutFab allowing them to get a more practical education.

Graz University of Technology (GUT) Institute of Innovation and Industrial Management has been operating a learning factory named LeanLab since 2014, which is focused on enhancing the university graduates' academic education, and offering external company training, in addition to industrial engineering and logistics hands-on

research (Karre et al.). GUT LF in their study identified short and long-term priorities, summarized Industry 4.0 upcoming challenges and created a roadmap for their LF.

Area	Technology
Communi-cation	Self-organizing communication network
	Real-time wireless communication
	Wire-based high performance communication
	IT-security (data protection, information security)
	Mobile communication channel
Embedded systems	Miniaturized embedded systems
	Energy harvesting
	Intelligent embedded systems
Human machine interface	Human behavior model
	Context-based information presentation
	Semantics-visualization
	Gesture-/ voice control
	Perceptual user interface
	Remote maintenance
	Virtual / augmented reality
	Intuitive operating elements
Sensor and actuator technology	Miniaturized sensors
	Intelligent and (re-)configurable sensors
	Sensor fusion
	Intelligent actuators
	Networked actuators
Standard-izations/Norms	Communications standards
	Standardization of system elements
	Identification standards
Software and systems technology	Simulation environment
	Multi-criterial situation evaluation
	Multi-agents systems
	Machine learning and pattern recognition
	Big data storing procedure and analytical methods
	Cloud computing
	Web- and cloud services
	Ontology

Figure 9 - GUT LF - Future Industry 4.0 Technologies (Karre et al.)

Industry 4.0 is focused on integrating the full information and communication technology stack including virtualization, clustering, cloud, cyber-physical systems, internet of things, and finally manufacturing as a Service. Many universities have contributed to part of the full information and communication technology stack; however, a complete Industry 4.0 ICT infrastructure research is yet to be found in the literature. TU Wein is building the I40PF focused on CPPS and they mention that they will have state-of-the-art information technology hardware infrastructure and software; however, they did not show or mention what are these IT infrastructure system components and how will they build it (Erol et al.). Similarly, PU HNI is building two LFs, i.e., PCP LF focused on Product LC and CPPS LF focused on SBC acting as CPS per production unit replacing the old PLC-based control systems (Gräßler, Pöhler, et al.; Gräßler, Taplick, et al.). In both universities above, the focus is mainly on Cyber-Physical Production Systems, which is just one piece of the big Industry 4.0 puzzle. On

the other hand, some universities, .e.g., RUB LPS LF, focus on Industry 4.0 learning modules in assistance with manual and CNC lathe and drilling machines, drilling tools, assembly station, etc., which is a great effort to add to the Industry 4.0 education map (Prinz et al.; Bender et al.); however, they do not show us how they intend to build an Industry 4.0 LF that should encompass not only automation but CPS, CPPS, CMfg, IIoT, MfgaaS and other Industry 4.0 related technologies.

Industry 4.0 Technologies

Cyber-Physical Systems (CPS) is the evolution resulting from the revolutionary advancement of interdisciplinary collaborating science disciplines, i.e., mechanical/mechatronics, computer science, electronics, and communication. CPS touches and affects nearly everyone's life on a daily bases through smartphones, smart cars, smart access, etc. CPS in very simple terms is the ability of the cyberinfrastructure, represented in sensors, computing power and communications hardware and software, to monitor intelligently our physical world and to control it using actuators and advanced controls systems. CPS consists of a strongly coupled cyber "computational" and physical subsystems that operate efficiently, dependable, safely and securely, which represents a mixture of embedded systems, distributed systems and real-time systems with advanced features like green energy efficient sensors, microcontrollers, networking and actuators (Quaglia, "Communications in Cyber-Physical Systems"; Rajkumar et al.). One way of defining CPS is to be an intelligent embedded distributed system with cyber and physical components operating to monitor, coordinate and control, using computing and communication, life-critical applications, where physical data captured through embedded systems and sensor networks, followed by actions enforced by actuators and management process and services (Wan et al.; Broy et al.).

CPS can be extremely life-critical context-aware networked system once associated with medical devices, which otherwise known as Medical CPS (MCPS). Safety, security, privacy, efficiency, and verifiability are few examples of the challenges facing CPS in medical science; nonetheless, CPS is widely used nowadays in hospitals providing high-quality real-time care for patients (I. Lee et al.). CPS examples extend to many applications from which is smart electrical grids, environment monitoring, supervisory control and data acquisition (SCADA), distributed control systems for automation networks, etc. (Al-Anbagi et al.; Antonini et al.). Consequently, CPS is the enabling technology for industry 4.0 and Internet of Things (IoT) allowing any individual and any physical object "thing" to communicate and exchange information and actions anytime and anyplace using any available network and/or service (Vermesan, O., Friess, P., Guillemin, P., Gusmeroli, S., Sundmaeker, H., Bassi, A., Jubert, I.S., Mazura, M., Harrison, M., Eisenhauer, M. and Doody).

Virtual/Cyber and Physical worlds have been having ongoing close parallel development, which is noticeable comparing Computer Science (CS), Information and Communication Technologies (ICT) and Manufacturing Services (MS) (Monostori et al.), which shown in below Figure 10. In other words, the development of one thing in the cyber/virtual world led to parallel development on the physical world. On the other hands, some researchers refer to CPS as Cyber-Physical Production Systems (CPPS) with the justification that CPPS relates more to production and manufacturing. It can be concluded that they use the same CPS architecture and references. Therefore, CPPS is nothing but a new notation with the same meaning of CPS and if encountered, it means nothing but CPS with more focus on production and manufacturing.

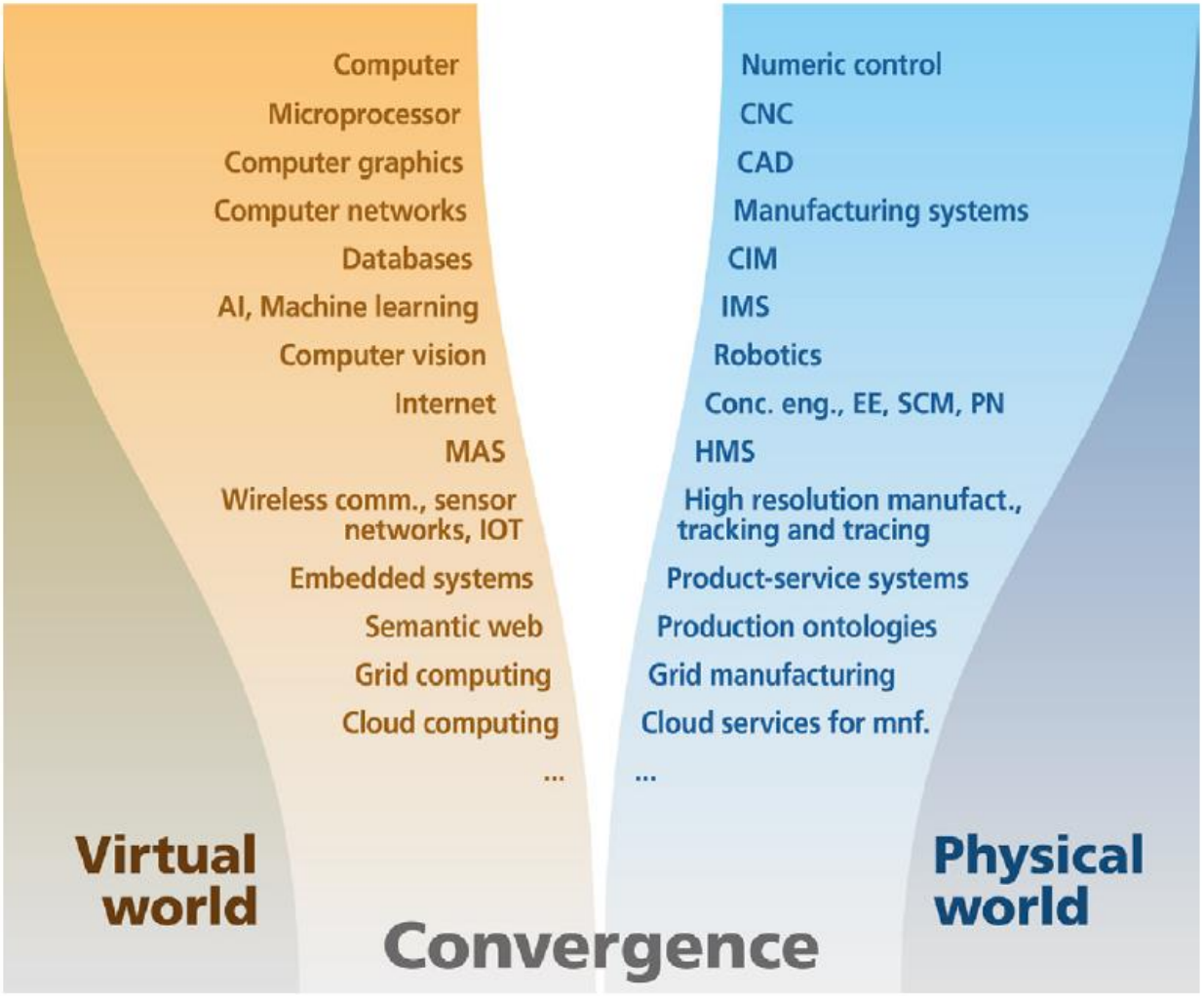

Figure 10 - Parallel Advancements in Virtual vs Physical Worlds (Monostori et al.)

Historically, CPS notation goes back to the first National Science Foundation (NSF) workshop held in Austin Texas on Cyber-Physical Systems in 2006. The conference webpage contained the following announcement: “The research initiative on cyber-physical systems seeks new scientific foundations and technologies to enable the rapid and reliable development and integration of computer- and information-centric physical and engineered systems. The goal of the initiative is to usher in a new generation of engineered systems that are highly dependable, efficiently produced, and capable of advanced performance in information, computation, communication, and control.

Sensing and manipulation of the physical world occur locally, while control and observability are enabled safely, securely, reliably and in real-time across a virtual network. This capability is referred to as Globally Virtual, Locally Physical". Additionally, the USA President's Council of Advisors on Science and Technology (PCAST) on a formal assessment of the Federal Networking and Information Technology Research and Development (NITRD) concluded to treat CPS as the top priority for federal research investment, which marked the starting of the CPS industrial competitiveness racing to innovation era (Challenge).

One of the main concerns of the industry when it comes to building a CPS is how resilient it will be. The three main important features of a resilient CPS are stable, secure and systematic. In a networked control CPS, sensors feed the inputs, computational logic makes the decisions and actuators execute the decisions. A stable CPS is the one that makes the right decision regardless of the environmental noises and interferences. Additionally, a secure system is the one that detects and counters all intrusion attacks cyber and/or physical. Lastly, a systematic CPS is the one designed with seamless integration of its cyber and physical components (Mitchell and Chen; Mo et al.; Hahn et al.; Hu et al.). Consequently, to achieve secure, stable and systematic CPS, the system faces challenges, i.e., adaptability, consistency, reliability, dynamics correspondence, and tight coupling.

The CPS needs to be able to adapt to behavioral changes in the system. The computation world has an intelligence running, which is receiving data from physical sensors, interpreting them and making decisions. This computational intelligence uses programmability, observability, and computability characteristics to achieve the quality

of service, fault-tolerance, and timeliness. In other words, CPS computational intelligence needs to achieve flawless actions regarding system resources allocation and failure detection and impact. Consequently, the more adaptable the system is going to be, the more dependable it will be (Ravindran and Rabby).

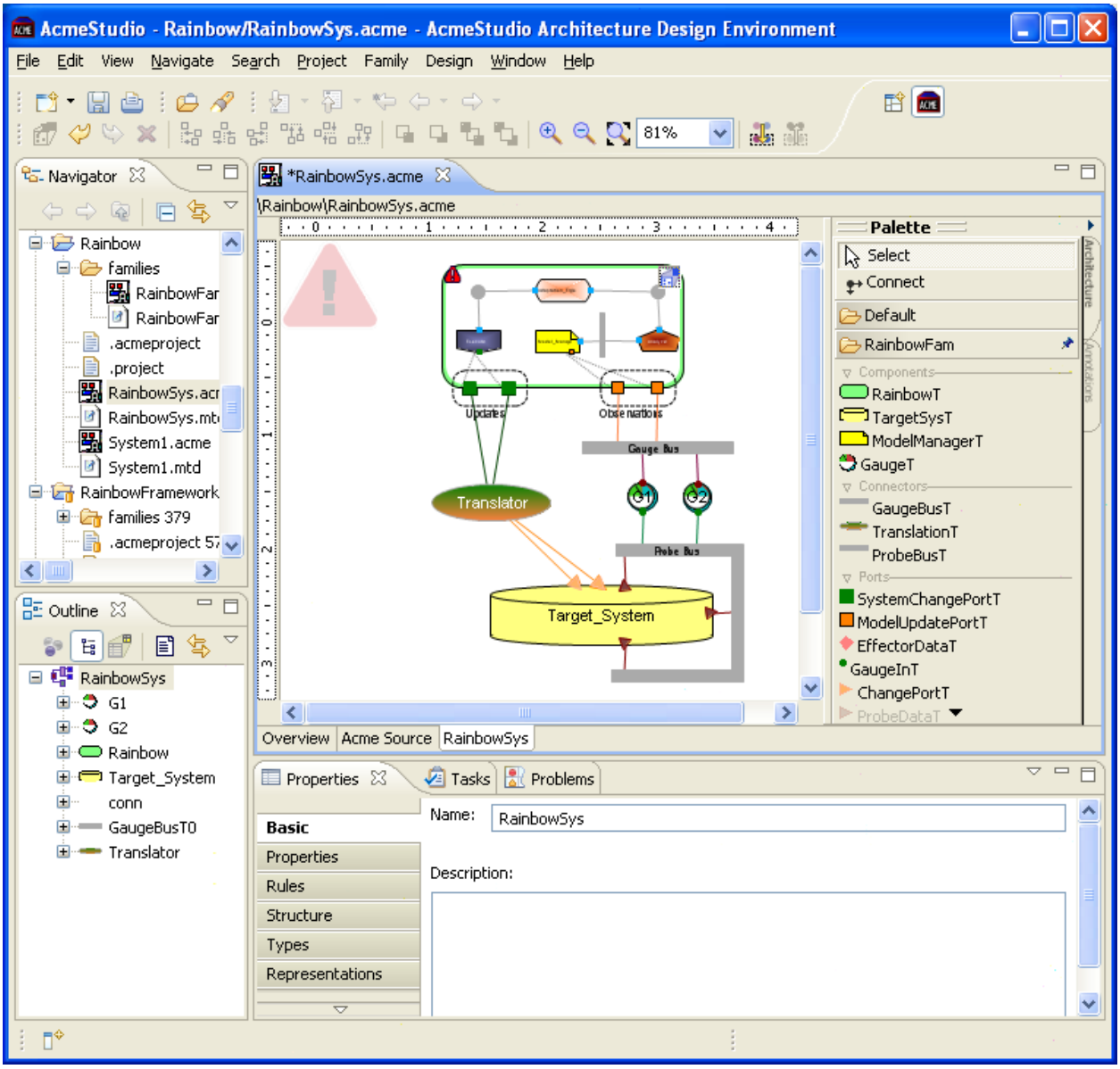

Figure 11 - Acme Studio Architecture Design Environment

Consistency is the ability of the CPS to account for all components in the Base Architecture (BA), i.e., all connections and communication paths between system

components is accounted-for, present and allowed in the BA. It is very important to have a detailed flawless architectural description, which is achievable using many software packages from which is Acme Studio Framework (Rajhans et al.). The Acme Studio checks for consistency in the BA with the system view through the component-connector graph. A multi-view editor allows corrections of inconsistent elements in the architecture/systems view. Figure 11 shows a view of the Acme Studio customizable editing environment and visualization tool used for software architectural designs using the Acme Architectural Description Language (ADL).

In the cyber world, a computer program execution is essentially perfectly reliable, as it will go through the same set of commands in the same order every time it runs. However, that is not the case when we involve physical objects in play, as there is always some uncertainty and delays that lead to unreliability. Essentially, there are a couple of ways to deal with uncertainty in computer systems by reducing the uncertainty through implementing error correction algorithms to overcome electronic/physical components errors (Rajamaki et al.). The ability to predict an error and be prepared for it is a great asset to any system achieved by system state early error prediction artificial intelligence and/or machine learning schemes, e.g., a hidden Markov model and/or regression model to predict following step system state relying on historical state data.

One of the main characteristics of the physical objects in the real world is the dynamic of the system, where the system state changes over time. However, that is not the case when we talk about the cyber world, as the dynamics simply does not exist since every action is a well-known sequence of code execution with no real temporal semantics. To help analyze the dynamics correspondence challenge, there are two methods involving

either calibrating cyber to physical interfaces or vice versa. The actions of imposing the cyber interface properties on the physical system and its opposing of software and cyber components calibration to dynamic real-time physical representation are so-called "cyberizing the physical" (CtP) and "physicalizing the cyber" (PtC) (Deshmukh et al.; Li and Li).

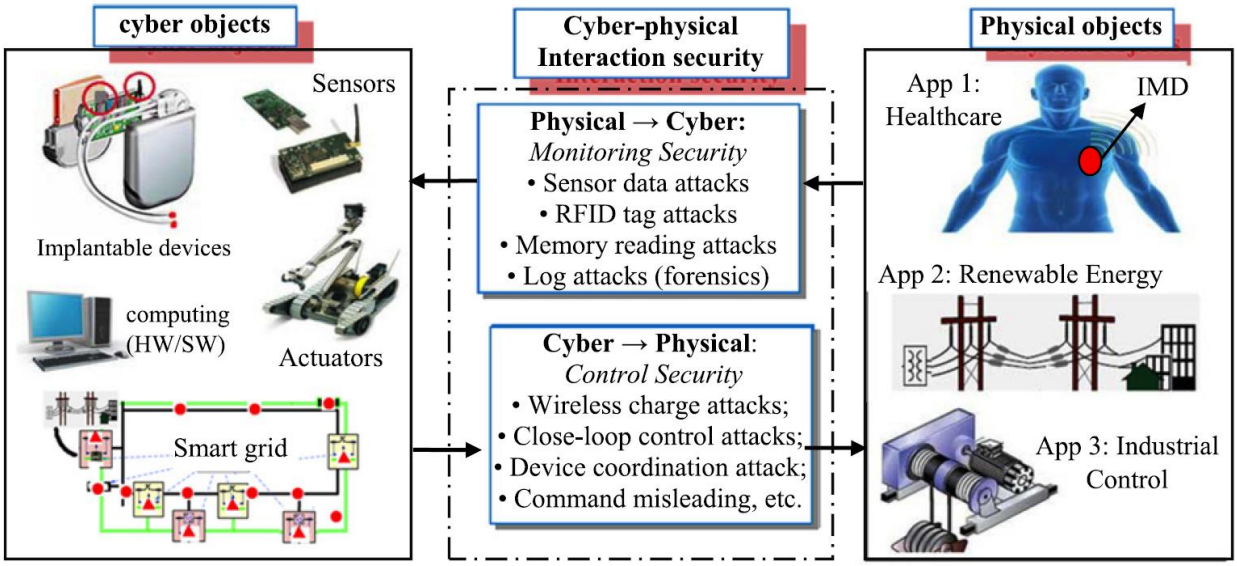

Figure 12 - CPS Security Interactions Perspective [43]

Tight coupling is very important when we deal with life interacting and affecting technologies that not only monitor, but also have the capability to alter the course of the future. This means that in order to implement a CPS into real-time real-life scenarios we must ensure that it is resilient and flawless. Resilience is important for both natural system faults and malicious system attacks. CPS needs advance cybersecurity schemes and anti-measurements to ensure its protection. In other words, all CPS security solutions have to have tight coupling and flawless integration within the underlying physical processes to control all CPS features (DeMarco et al.; Cárenas and Amin). An example of a CPS that is human life affecting is the Implantable Medical Device (IMD),

which is implantable in a human body for medical sensing and monitoring, and for organ control, e.g., pacemakers, insulin pumps, neurostimulators, etc. The above Figure 12 shows the CPS interactions with security.

Manufacturing industry withheld outstanding advancements recently paving the way for the Fourth Industrial Revolution, i.e., Industry 4.0, led specifically by Cyber-Physical Systems. CPS is playing the key role in the future engineering systems design and development capabilities in cybersecurity, autonomy, reliability, functionality, and usability (Zanero). Such advancements in CPS is due to the collaborations between governments, industry and academic disciplines in control, communication, computation, and computer science. Consequently, sensors, actuators, data acquisition systems, computer networks, etc. are highly available, affordable and easily integrated into today's manufacturing, which applies a hidden strong force on all factories and industries to implement and invest in CPS and Cloud Manufacturing. With this advancement and huge investments put into CPS, a new generation of highly useful and high volume of data come into manufacturing, known as Big Data. Thus, manufacturing nowadays is shifting to embrace not only CPS but also Big Data and Analytics to improve the whole manufacturing cycle. Therefore, CPS is the key to future manufacturing as it opens the door for all those new technologies implementations at the Smart Factory of the future, i.e., Industry 4.0 Factory.

CPS tightly couple the cyber and physical resources in the manufacturing, telecommunications, and IT. The tightly coupled model follows the systematic "5S" approach consisting of Sensing, Synchronization, Storage, Synthesis, and Service. It operates on the cloud platform, where it collects and stores the data, as well as applies

analytical algorithms. Thus, Prognostics and Health Management (PHM) future manufacturing model is high in demand with the current critical research leveraging the advanced predictive tools. The PHM implementations will feed off the data collected during the manufacturing life cycle leading to better machining health and overall manufacturing (J. Lee, Lapira, et al.).

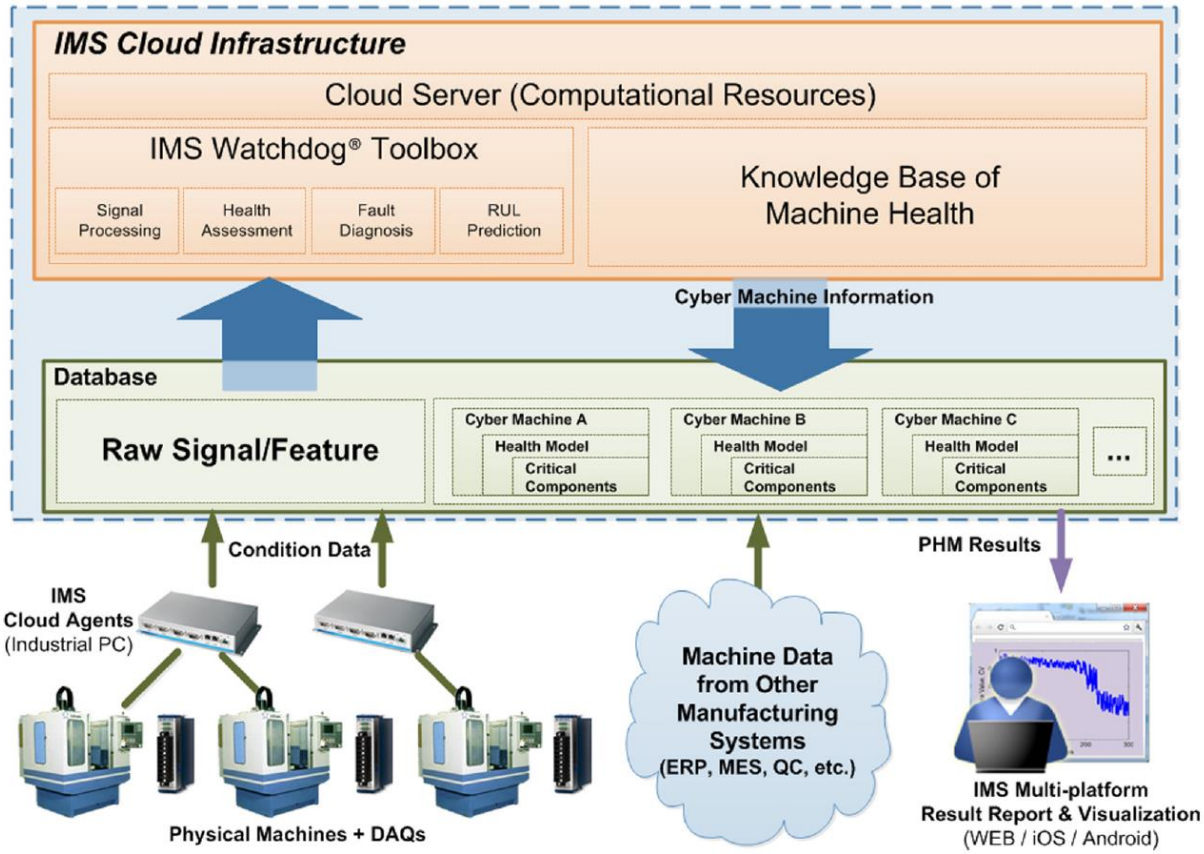

Figure 13 - CPS Model for Enhanced Predictive Manufacturing System (J. Lee, Lapira, et al.)

The above five level CPS architecture acts as a detailed guideline for CPS Manufacturing development and deployment. CPS is breakable into two key functional components connectivity and computing. First, the cutting-edge connectivity ensuring real-time data acquisition from physical to cyber worlds, and informative/actionable feedback from cyber to physical worlds. Second, the computer advancements

represented in seamless data management, analytics, and computational capability. Yet, with such understanding of key functionalities, a clear implementation workflow for constructing CPS is crucial to successfully adopt the technology. The following 5C architecture shall define a distinct way of sequentially designing CPS workflow implementation (J. Lee, Bagheri, et al.).

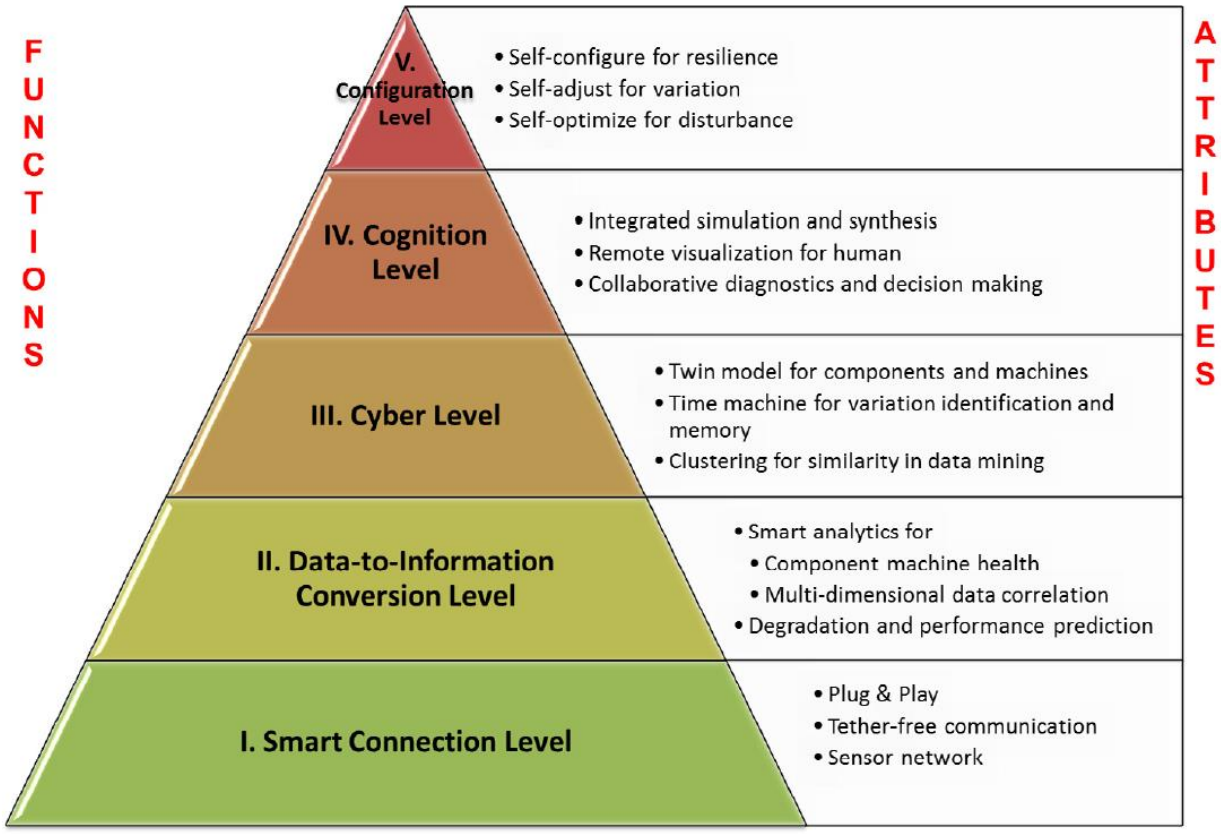

Figure 14 - CPS Implementation "5C" Architecture (J. Lee, Bagheri, et al.)

Smart connection level focuses on accurate and reliable data acquisition from different components of the CPS. Connections occur on many levels for a variety of data type acquisitions, e.g., tether-free connections to sensors and/or controllers, application connections to enterprise manufacturing systems such as ERP, MES, SCM, and CMM, etc. Consequently, tether-free connections mean that CPS have to connect to either

wide-range cellular, e.g., GSM, UMTS, CDMA, LTE, etc., or wide-range non-cellular, e.g., Wi-Fi. On the other hand, connection to the enterprise manufacturing systems achievable in many ways according to the hosting of the system database and it is much easier to configure. Additionally, connections mean that we have channels open between to entities, which indicates security vulnerabilities. Therefore, data connections design must ensure high-security solutions, e.g., data encryption, tunneling, virtual private networks, etc. (J. Lee, Bagheri, et al.).

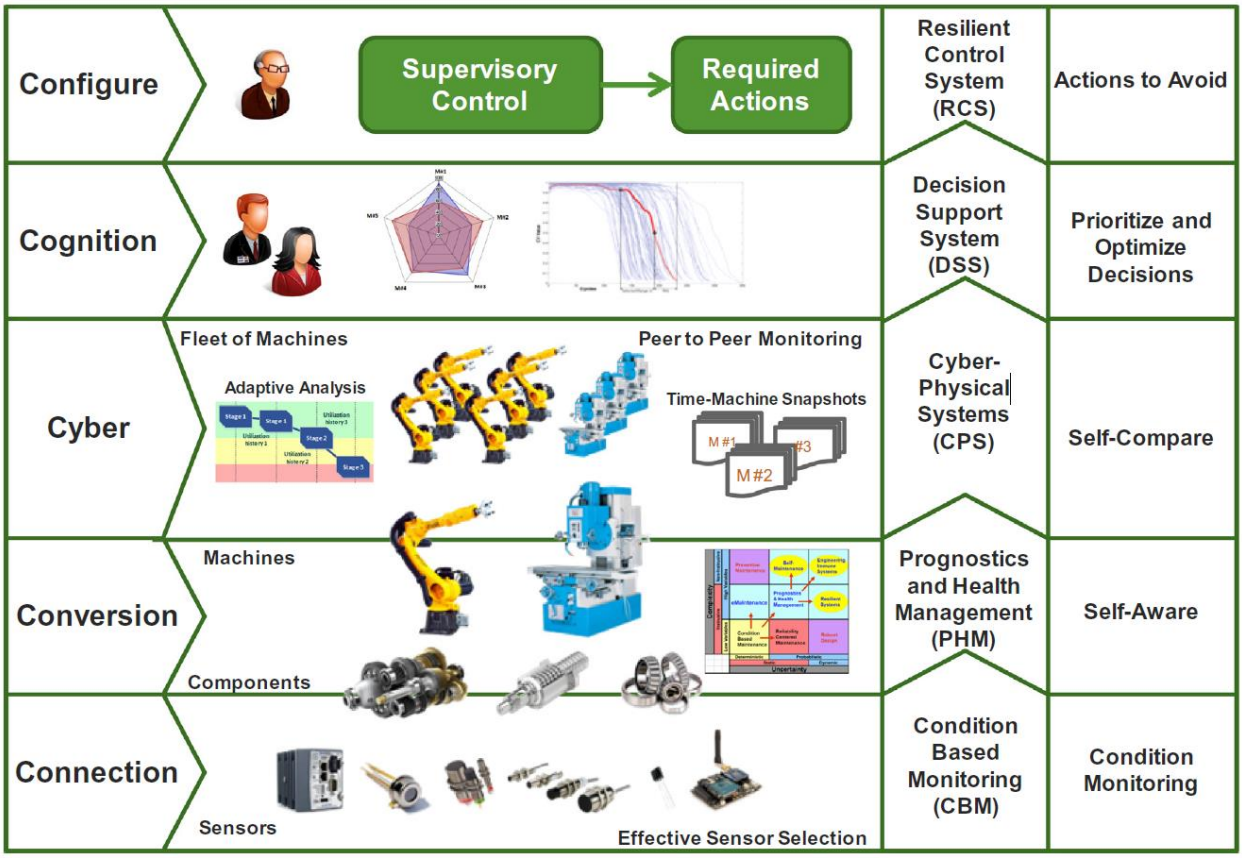

Figure 15 - CPS Implementation "5C" Architecture Applications and Techniques (J. Lee, Bagheri, et al.)

Data-to-information conversion level is all about extracting or converting raw data into meaningful information. Information fed back into the system to improve the overall process and to generate new revenue streams based on the analytical feedback and

statistical data. There are many levels of enriching raw data to become useful information and it happens on many levels of the system. Data enrichment could happen at the collection level on the microchip attached to the sensor, at the server levels once the data uploaded to the cloud, or at the analytics platform. Each data enrichment method has a cost associated with it and each CPS designer must choose the best possible scenario to ensure real-time seamless operations of the entire system with the maximum value and performance of all inputs and outputs (J. Lee, Bagheri, et al.).

The cyber level is the place were everything in CPS converge and diverge. The convergence happens on the data acquisition from all different connections followed by data storage and analytics engine working in the background creating meaningful information. The divergence occurs once the CPS has feedback data or actionable data to feed back into the physical space. The cyberspace in the context of Industry 4.0 exists in the cloud through any of its versions, i.e., private, public or hybrid, were compute, storage and bandwidth are unlimited with pay-as-you-go model facilitating meeting the new technologies high volume data, high compute and larger bandwidth demands (J. Lee, Bagheri, et al.).

Cognition level emphasis accurate acquired knowledge presentation to systems administrators, experts, and executives facilitating decision making and road map plotting. It is crucial to be able to come up with statistics, charts and informative graphics to ease the knowledge transfer and decision-making processes. Decision-making can be manual, semi-automated or fully automated according to the level of investment and applications in manufacturing.

Configuration level is the cyberspace feedback to physical space enabling the connected machines to be self-configurable, self-adaptive, self-optimizing and self-adjustable. This level behaves as a Resilience Control System (RCS) applying cyberspace analytics generated corrective and preventive decisions (J. Lee, Bagheri, et al.).

Modeling is very crucial to any system especially CPS, as it can demonstrate the design process evolution of the system and help specify the governance, administration, and specifications of the desired system (Zhai et al.). Modeling creates this virtual safe environment providing engineers the capability to test, enhance and avoid design defects before implementation into real life (Ma et al.). It is of great importance to understand how to model CPS especially timing and scheduling of various actuation events and how to reflect on the action interlocking relationship, and to understand different types of security and control in CPS model with quantitative cyber and physical interaction (Quaglia, "Cyber-Physical Systems: Modeling, Simulation, Design and Validation"; Moreno et al.; Bu et al.). CPS is similar to embedded systems in the manner that physical processes are made-up of a combination of other parallel running processes, where the CPS is orchestrating, measuring and controlling actions and interactions between those processes (Gao et al.). In order to model CPS, engineers need to build several models for physical and cyber spaces specifically software, computational platforms, networks and physical resources (Cheng and Chang).

The IoT is now perceived as the most common model available that can bring the cyber and physical worlds together. It is crucial for such a paradigm to follow in the green environment-friendly technology path as it is growing to be one of the most widely spread technologies adopted in nearly all industries and homes. This noticeable growth

comes with a cost of challenging complexity, especially when deploying Wireless Sensor Networks (WSN). In (Huang et al.), the authors are trying to find an optimization model based on the generic green hierarchal structure and then they propose a solution for the optimization model. They have proved numerically that the proposed algorithm in this paper has a better performance than the typical WSN deployment scenarios. In the introduction, the authors are defining the IoT to be the network consisting of smart objects. These smart objects vary from Radio Frequency Identification (RFID) tags, sensors, and actuators, which are associated with network either wired or wireless. Those smart objects are used in monitoring and controlling the physical world enabling individual in the cyber world to have real-time data of their physical objects, thus better and swifter resource management.

The advancement and widespread use of IoT introduces a lot of complex challenges, which are not present in the current WSN models. Since IoT is transforming all industries and widely spreading to incorporate most if not all smart objects; in other words, IoT is becoming the biggest, most complex and widely spread network. Therefore, IoT objects combined uses tremendous amounts of energy, thus must be regulated with green cost-efficient networking deployments, which are the main focus of the researchers. In (Huang et al.), the researchers argue that most energy efficient deployments of WSN are not conducted with the consideration of the green networking paradigm that could serve as a highly scalable and sustainable IoT. They introduce how to cost-efficiently deploy or implement green networked IoT. They introduce the hierarchy of the deployment, followed by an optimization model, and, eventually, an algorithm to solve the optimization problem. The algorithm is based on clustering and Steiner Trees, and they call it the minimal energy consumption algorithm (MECA). In

Chapter 5, we shall introduce an Enhanced MECA (EMECA) model that shall take into consideration the recent advancements in the WSN models leading to a less complexity provided algorithm. In addition, we shall implement the code for both MECA and EMCA showing both visual and textual results proving our enhanced model less complexity in terms of using the Industry 4.0 digital foundation design less complex chipsets on the sensing nodes transferring the computational power, decision making and feedback loop to the private cloud; thus, enabling less power consumption, minimizing overall budget and improving data link transfer.

The Industrial Internet of Things (IIoT) or in easier terms the industrial network of things is the integration between physical devices specifically in an industrial environment and computerized systems through a well-established network resulting in an improved overall production efficiency and accuracy leading to clear economic profit with reduced human intervention (Samad). Conventionally, IIoT applications are hosted on a central computing facility, e.g., Public Cloud, using centralized data processing model, where data generated by industrial devices is transported over the Internet infrastructure to the public cloud in order to get analyzed and intensively processed. However, this growing demand for seemingly intensive data over the growingly over congested Internet leads to unpredictable network slowness and latency. This critical alarm that data transferred and processed over the Internet is not going to meet the Service Level Agreements (SLAs) requirements necessary for monitoring, reporting, and controlling critical industrial applications, devices and production lines (Suto et al.).

In order to overcome the Internet congestion limitations, the above architecture introduces the Fog Computing, which has recently been integrated into IIoT

architecture. Fog Computing supports real-time response and high automation exploiting the spare computing resources of edge devices, e.g., network gateways, enabling ultra-low latency responses, i.e., unburdening the Internet traffic backbone (Lin and Yang). Commonly, industrial production line devices are equipped with smart Sensor Nodes (SNs) which collect data and perform first order operations, e.g., filtering, aggregation, and translation, on the raw data. In a study, a group of SNs is logically clustered to communicate with a single Fog Node (FN) enriching computing resources provided. In terms, the FNs receiving streamed data from SNs perform complex data analysis and derive actionable intelligence maintaining the industrial production devices within a sphere of influence with the optional offloading of additional workloads exceeding the FNs influence sphere computing capacity to the Public Cloud resulting in a Hierarchical Fog Computing Architecture (Azimi et al.; Chen et al.).

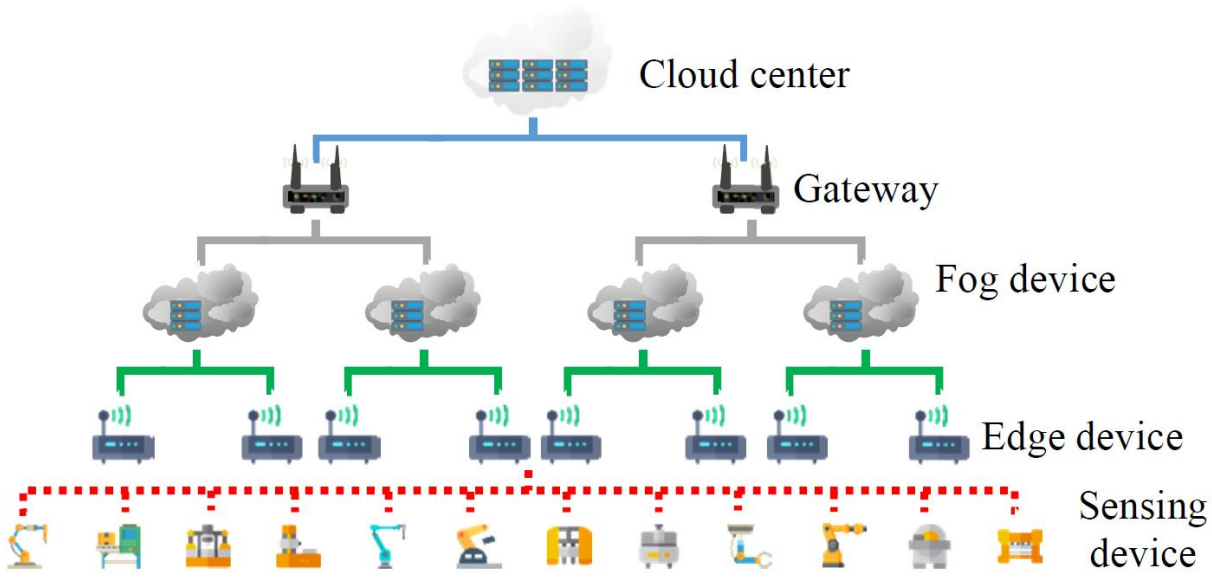

Figure 16 - The Architecture of Fog Computing (Lin and Yang)

The Task Offloading is not a new concept as it has been a key component in many other integrations concerned with what, when and/or how to offload workloads off devices to

FNs (Xu et al.). Unlike the Public Cloud expandable pool of on-demand resources, the FNs has a small set of limited computing and storage resources sufficient only for limited simultaneously hosted on-demand services (Yang et al.). Generally, configuring Fog Networks and Nodes is an extremely challenging problem for the IIoT since industrial production devices vary in functionality and services, thus, producing different datasets to be sensed and analyzed. In other words, matching and maintaining FN accompanied SNs and hosted services while maintaining Fog Computing overall performance and efficiency is a rather heterogeneous, difficult and highly complex task. IIoT is a rather new concept that is going to have more innovation and one feature that would rather act as a supportive if not replacement to the Fog Architecture is having Hybrid Cloud implementations on all industrial sites, where local onsite Private Cloud could act as the immediate data gathering and processing unit offloading to the Public Cloud the excessively computational tasks that require more resources.

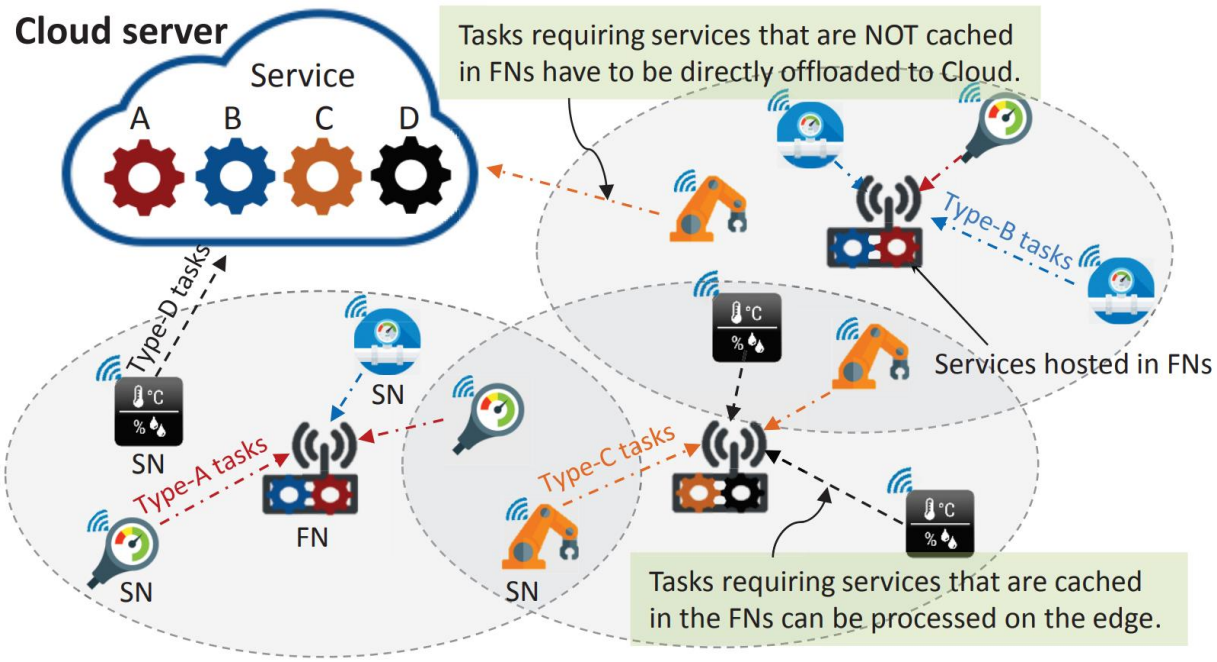

Figure 17 - Adaptive Fog Computing Configuration Illustration (Chen et al.)

Cloud Manufacturing offer organizations the competitive edge needed in today's fast-paced, online, and highly interconnected global economy allowing them to be agile, flexible, and able to respond rapidly to the changing market conditions. Cloud, a next-generation style of computing, provides highly scalable and flexible computing that is available on demand. Cloud Manufacturing empowers self-service requesting through a fully automated request-fulfillment process in the background. It promises real costs savings and agility to organizations.

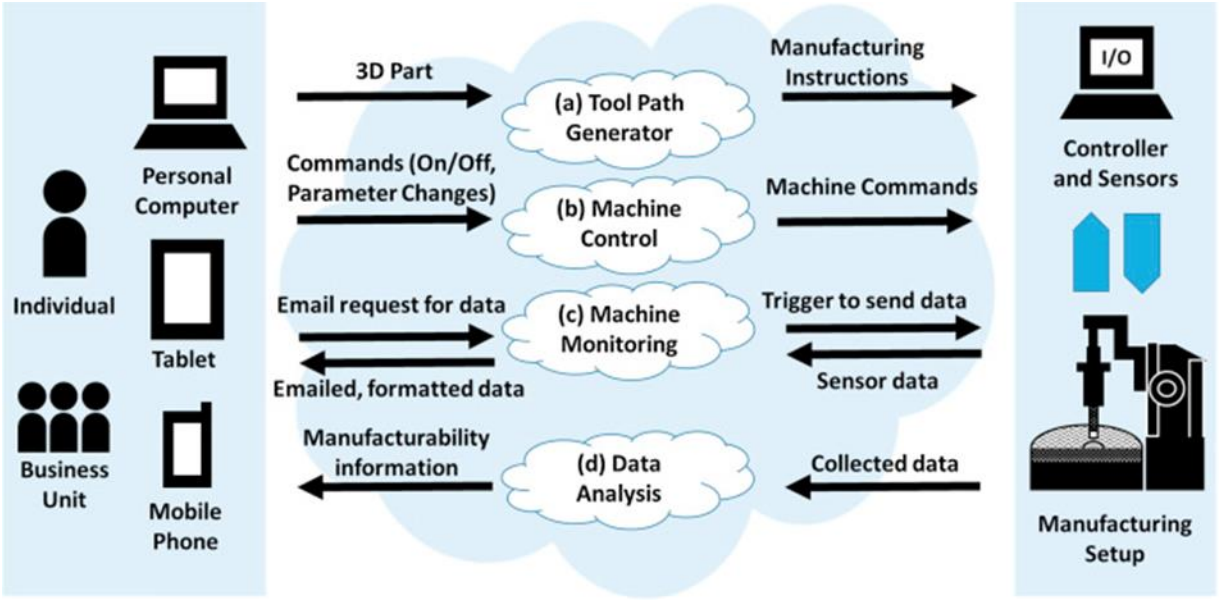

Figure 18 - Cloud Manufacturing Architecture (Brant and Sundaram)

Through Cloud, an organization can rapidly deploy applications where the underlying technology components can scale-up and scale-down, based on the business requirements. In other words, linking manufacturing, specifically additive manufacturing, to the cloud leverages the power of constant remote real-time communication, monitoring, and control with any user phone, tablet or personal computer (Brant and Sundaram). Additionally, manufacturers now only pay for what they use and can scale their cloud usage on demand anytime for any period.

Within the past few years, the interest in Cloud technologies overwhelmed many industries. With this interest surrounding Cloud, it is helpful to understand what the driving forces behind it are. Each organization has its own unique drivers, which falls into some general categories, e.g., cost, availability, time-to-market, etc. Mainly management exerts tremendous pressures on IT from the organization's highest-level executives who are looking for more value, doing more with less cost, and using information as a competitive edge. Cloud offers the transformation needed by IT in terms of business agility, scalability, less cost and more efficiency. All in all Cloud Computing has proven in many studies to have better cost-effectiveness over older technologies (Kondo et al.). Organizations face many IT challenges, which drove them to consider and, shortly after, adopt the Cloud Computing model. Globalization challenge, where IT must meet the business needs to serve customers worldwide and around the clock. The aging of data centers and the cost of migrations, upgrades, and replacements of old technologies. In addition, the explosion of storage consumption from generated data, which entails the necessity for storage growth. More importantly, cost of ownership challenges that surface due to increasing business demand, the cost of buying new equipment, power, cooling, support, licenses, etc., increases the Total Cost of Ownership (TCO) (Truong and Dustdar). Cloud infrastructure should fulfill the essential characteristics to support Cloud services. One way is build it using a shared pool of computing resources, such as compute, storage, and network. The infrastructure should be flexible to meet the rapidly changing demands of its consumers and allow them to provision resources on-demand over a network. The infrastructure should also enable monitoring, control, and optimization of resource usage. Consequently, building a Cloud infrastructure is a phased approach. The journey begins with understanding the

existing physical infrastructure, its elements, and processes. The next step is to focus on aggregating the existing infrastructure resources using virtualization technologies. These resource pools facilitate centralized management of resources and enable faster resource provisioning (Peng and Jiang). Cloud Services realized through a service management tool integrated on top of the virtualized infrastructure. The service management tool enables the creation and optimization of Cloud services to meet business objectives and to provide value to the consumers. The services built enlisted in a service catalog that allows consumers to choose the desired services. Additionally, service management automates service creation and provisioning without any manual intervention. It also helps the monitoring and metering services in measuring resource usage and chargeback. Service management tools are also responsible for managing both physical and virtual resources used to create Cloud services. Examples of management activities are capacity management, configuration management, change management, etc. These management processes enable meeting service assurance and compliance requirements (Lenk et al.).

In general, there is a need to have a complete integration between CPS, CPPS, IoT, IIoT, and Cloud Manufacturing revealing the true potential behind the Industry 4.0 technologies and highlighting the strengths in them. Most literature focuses on a single topic with a quick snap of the overview and integration between Industry 4.0 technologies. However, in this work, we try to fill in some gaps in the literature Industry 4.0 intercommunicating technologies going from as low level as the h/w and sensors and all the way to high-level services for not only consumers but also decision makers, engineers, and executives.

CHAPTER 3 INDUSTRY 4.0 FUNDAMENTALS

Industry 4.0 implements the Service Oriented Paradigm, where manufacturers leverage the advancements in ICT industry to not only improve their industrial footprint but also to establish stronger more profound connections with customers in terms of providing them with a new set of applications and services ensuring better customer engagement, higher satisfaction, enhanced quality of service and new customer-driven revenue streams. Such ICT enabled system design contains multiple levels of local and remote hardware and software design components. The first step on the path to implementing Industry 4.0 ICT digital foundation is to transform the existing IT infrastructure to the cloud architectural model, preferably a hybrid cloud model, which has both private and public components. The cloud model enables the deployment of infrastructure as a service, platform as a service and software as a service, which facilitates hosting software solutions to monitor, control, virtualize and customize production facilitating providing subscription and customization services to the customers.

Breaking down the on-premise local ICT infrastructure, the base component is the local cloud, which is either a high availability cluster of virtualized server racks with cloud management software on top or cloud ready rack-scale hyper-converged solutions. Virtualization is established on the storage, compute and network sublevels, while cloud management tools are used to establish service levels, metering, and service distribution and management. An example of how the cloud ICT infrastructure enables Industry 4.0 technologies is the cloud service offering Infrastructure as a Service, which is a key enabler to the cyber-physical systems to provide monitoring, reporting, control, and optimization. Additionally, some CPS software is already developed to be cloud-

enabled, which means it is fully integrated with the cloud platform to provide Software as a Service to the specific software enabled manufacturing CPS.

The Industry 4.0 most important piece of the Information and Communication Technology puzzle is the software infrastructure component, which in terms is nothing but the web and mobile service offerings developed to provide a generic platform for monitoring, reporting, actuation, customization and subscription services. The web and mobile application modules providing the services act as the front end of the entire software solution package, which feeds off a backend backbone software service connected to the database. In general, the base software is nothing but a generic three-layered application containing frontend, backend, and database, which acts as a facilitator to and from manufacturers and customers. The backend is the core component of the software as it is the main computation and querying part, where it acquires and feeds data and instructions to and from both the CPSs forming the production line and the database. The backend saves all the data on the database and retrieves data according to the frontend demand of designers, engineers, managers, and customers.

A simplifying example of a distributed Internet of Things Industry 4.0 lookalike system is the university-wide printing service, where anyone with valid credentials can use the printing service website to log in, add money to his account, upload documents, and finally go to any printer on campus and print his document. Industry 4.0 takes this explanatory printing example to a whole new level in production with much more complex subsystems linking not only machine and production lines across countries but also customers, retailers, suppliers and service providers. The fourth industrial revolution is about building blocks of infrastructure, consequently, in the below few

sections we shall focus on the key concepts and main information technology pillars upon which the building blocks of the fourth industrial infrastructure lies.

Information Technology Fundamentals

Data

Data by definition is a collection of raw facts, which could be used to draw conclusions. Data is found in everything around us from letters, books, photographs, papers, ledgers, tickets and much more examples from day-to-day life. Data creation and sharing were limited to paper and film before the invention of computers, which marked a transformation of data sharing, storage, and creation, e.g., emails, e-books, digital photos, digital movies, etc. Computer generated data is stored as strings of binary numbers, i.e., zeros and ones, which is referred to as digital data.

Digital Data is only accessible to users who possess a computing device to process it into a meaningful representation. Businesses rely heavily on digital data to analyze the vast raw data they generate on daily basis. Meaningful trends and insights are drawn from raw digital data identifying weakness and strength improvement areas that help organizations and businesses plan, strategize, finding new business opportunities and generating new revenue streams. Digital Data growth rate has been increasing exponentially due to the vast spread of applications and smart devices, increase in data processing capabilities, low cost of digital storage, and the surge of fast affordable communication technology.

Based on data storage and management, it is classified as either structured or unstructured. If data could be organized in rows and columns, then it is denoted as structured data, which is stereotypically stored in a database management system

(DBMS), where it could be easily retrieved and processed. On the other hand, unstructured data cannot be stored in rows and columns, which makes it difficult to query and retrieve by applications. The challenge is that the vast majority of new data being generated daily is unstructured yielding the industry in the continuous search for new techniques, technologies and architectural skills to store, manage, analyze, and derive meaningful comprehensions from unstructured data.

Big Data

Big data is a new growing concept referring to digital data sets with sizes far beyond common software tools capability to capture, store, manage, and process within acceptable time boundaries. Big data is comprised of both structured and unstructured data, which is generated by various types of sources, e.g., social media, banks, news, traffic, ports, etc. The big data sets challenge is that they should be real-time captured, updated, analyzed, modeled, and fed into decision making, which is inadequate to handle in terms of volume, variety, dynamism, and complexity by traditional information technology infrastructures and data processing tools and methodologies. Thus, the compelling need for new techniques, tools, and architectures to provide high-performance big data sets analysis using massive parallel processing (MPP) data platforms, and advanced analytics.

Big Data constitutes a huge part of the fourth industrial revolution as it is the area where all the data floating from all the connected Internet of Things devices converge. Scientists focus on finding new ways to cultivate this data with the backing up of the world’s governments, educational institutes, organizations and many more, which are spending tremendous amounts of resources on Data Science (DS) and Artificial

Intelligence (AI) disciplines in a sole focus of deriving business value from collected overwhelming big data sets. Both AI and DS represent the synthesis of many interconnected disciplines of statistics, math, data visualization and computer science enabling scientists to develop advanced algorithms capable of analyzing the massive amounts of information in order to construct a more valuable and data-driven decisions affecting broad areas, e.g., medical and scientific research, healthcare, public administration, fraud detection, social media, banks, insurance companies, and other digital information-based. This progression of Big Data research and technological advancement is all made possible due to the simplicity, efficiency, and inexpensiveness of the storage architecture required by big data facilitating access to multiple platforms and data sources simultaneously.

Storage

Data created by individuals or businesses must be stored so that it is easily accessible for further processing. In a computing environment, devices designed for storing data are named either storage devices or simply storage. The type of storage used varies based on the type of data and the rate at which it is created, stored and retrieved for usage. Examples of storage devices are media-cards/flash-drives in cell phones or digital cameras, DVDs, CD-ROMs, and disk drives in the personal computer. Similarly, businesses have several options of storage devices available for storing data, e.g., internal hard disks, external disk arrays, and tapes.

Historically, organizations had centralized computers, i.e., mainframes, and information storage devices, i.e., tape reels and disk packs, in their data center. The open systems evolution made it possible for business units/departments to have their own servers and

storage because of the affordability, and ease of deployment. In earlier implementations of open systems, storage was typically internal to the server, where these storage devices could not be shared with any other server. This approach is referred to as server-centric storage architecture, where each server has a limited number of storage devices, and any administrative tasks such as maintenance of the server or increasing storage capacity might result in the unavailability of information. As a result to the server-centric approach and the spread of departmental servers in an enterprise, the storage and servers became unprotected and unmanaged turning into fragmented islands of information and leading to an increased capital and operating expenses. To overcome these challenges, storage evolved from server-centric to information-centric architecture, where storage devices are managed centrally and independently of servers. These centrally-managed storage devices are shared with multiple servers, wherein the deployment of new servers in the environment storage is assigned from the same shared storage devices to that server. The capacity of shared storage can be increased dynamically by adding more storage devices without impacting information availability leading to an easier and cost-effective information management.

Storage technology and architecture continue to evolve, which enables organizations to consolidate, protect, optimize, and leverage their data to achieve the highest return on information assets. In fact, storage is the core component in a data center, where storage device uses magnetic, optic, or solid state media. Disks, tapes, and diskettes use magnetic media, whereas CD/DVD uses optical media for storage. Removable Flash memory or Flash drives are examples of solid state media. In the past, tapes were the most popular storage option for backups because of their low cost; however, tapes have various limitations in terms of performance and management, from which is the fact

that data stored on the tape is stored linearly along the length of the tape. Consequently, search and retrieval of data are done sequentially, which means that data retrieval can take invariably several seconds to access the data. As a result, random data access is slow and time-consuming, which limits tapes as a viable option for applications requiring real-time, rapid access to data. In addition, in a shared computing environment, data stored on tape cannot be accessed by multiple applications simultaneously, restricting its use to one application at a time. On a tape drive, the read/write head touches the tape surface, so the tape degrades or wears out after repeated use. The storage and retrieval requirements of data from the tape and the overhead associated with managing the tape media are significant. Thus, due to these limitations and availability of low-cost disk drives, tapes are no longer a preferred choice as a backup destination for enterprise-class data centers.

Optical disc storage is popular in small, single-user computing environments. It is frequently used by individuals to store photos or as a backup medium on personal or laptop computers. It is also used as a distribution medium for small applications, such as games, or as a means to transfer small amounts of data from one computer to another. Optical discs have limited capacity and speed, which limit the use of optical media as a business data storage solution. The capability to write once and read many (WORM) is one advantage of optical disc storage. A CD-ROM is an example of a WORM device. Optical discs, to some degree, guarantee that the content has not been altered. Therefore, it can be used as a low-cost alternative for long-term storage of relatively small amounts of fixed content that do not change after it is created. Collections of optical discs in an array, called a jukebox, are still used as a fixed-content storage solution. Other forms of optical discs include CD-RW, Blu-ray disc, and other variations of DVD.

Disk drives are the most popular storage medium used in modern computers for storing and accessing data for performance-intensive, online applications. Disks support rapid access to random data locations. This means that data can be written or retrieved quickly for a large number of simultaneous users or applications. In addition, disks have a large capacity. Disk storage arrays are configured with multiple disks to provide increased capacity and enhanced performance. Traditionally, high I/O requirements of an application were met by simply using more disks. Availability of enterprise-class flash drives (EFD) has changed the scenario. Flash drives, also referred as solid state drives (SSDs), are new generation drives that deliver the ultra-high performance required by performance-sensitive applications. Flash drives use semiconductor-based solid state memory (flash memory) to store and retrieve data. Unlike conventional mechanical disk drives, flash drives contain no moving parts; therefore, they do not have to seek and rotational latencies like disk drives. Flash drives deliver a high number of IOPS with very low response times; additionally, being a semiconductor-based device, flash drives consume less power, compared to mechanical drives. Flash drives are especially suited for applications with small block size and random-read workloads that require consistently low response times, i.e., less than one millisecond. Applications that need to process massive amounts of information quickly, such as currency exchange, electronic trading systems, and real-time data feed processing, benefit from flash drives. Overall, flash drives provide better total cost of ownership (TCO) even though they cost more on $/GB basis. By implementing flash drives, businesses can meet application performance requirements with far fewer drives approximately 20 to 30 times less number of drives compared to conventional mechanical drives. This reduction not only provides savings in terms of drive cost, but also translates to savings for power, cooling,

and space consumption. Fewer numbers of drives in the environment also mean less cost for managing the storage.

Data Center

Enterprises maintain data centers to provide centralized data-processing capabilities across all departments. Data centers infrastructure includes hardware components, e.g., storage systems, computers, network devices, and power backups; and, in addition, software components, e.g., applications, services, and management software. It also includes environmental controls as air conditioning, fire suppression, and ventilation, and also security controls, devices, and personal to safe keep the enterprise data and knowledge. Large organizations often maintain more than one data center to distribute data processing workloads and provide backup in case of a disaster. Data centers are made up of five core elements essential for their functionality, which are applications, databases, servers, network, and storage. Applications are computer programs that provide the logic for computing operations. Databases, mostly database management systems (DBMS), provide a structured way to store data in logically organized tables that are interrelated. Servers, also known as hosts or computer, are computing platforms made up of hardware, firmware, and software, which runs applications and databases. The network provides a data path that facilitates communication among various networked devices. Storage, as discussed above, is a collection of networked devices that stores data persistently for subsequent use.

It is critical to the survival and success of a businesses to maintain an uninterrupted operation of data centers, which is achievable through sustaining availability of information when required, security through establishing policies, procedures, and core

element integration to prevent unauthorized access to information, scalability without interrupting business operations deploying more servers based on requirements, data integrity ensuring data is stored and retrieved exactly as it was received, consistency optimal performance based on the required service levels, and manageability providing easy and integrated management of all its core components through automation and reduction of human intervention in common tasks. Managing data center involves many key activities and tasks such as monitoring through continuous process of gathering information on various elements and services running in a data center, reporting on resource performance, capacity, and utilization helping to establish business justifications and chargeback of costs associated with data center operations, and provisioning through providing the hardware, software, and other resources required to run a data center.

Data center infrastructure resources have dramatically changed the way they are provisioned and managed after the introduction of virtualization and cloud computing. Organizations are rapidly deploying virtualization on various elements of data centers to optimize their utilization and on top of it the cloud computing to better meter, provision and forecast resources.

Virtualization

Virtualization in data centers is a technique of abstracting physical resources, such as compute, storage, and network and making them appear as logical resources enabling pooling of physical resources and providing an aggregated view of the physical resource capabilities, which is nothing but the centralized management of pooled resources. In other words, pooling storage devices to appear as a single large storage entity is storage

virtualization. Similarly, compute virtualization is the aggregation of the power of all CPUs capacity of the pooled physical servers as one. Virtual resources can be created and provisioned from the pooled physical resources, e.g., a virtual disk of a given capacity can be created from a storage pool or a virtual server with specific CPU power and memory can be configured from a compute pool. These virtual resources share pooled physical resources, which improves the utilization of physical IT resources. Based on business requirements, capacity can be added to or removed from the virtual resources without any disruption to applications or users. With the improved utilization of IT assets, organizations save the costs associated with procurement and management of new physical resources. In addition, virtualization utilizes the most of any data center unused resources resulting in fewer physical resources meaning less space and energy, which leads to better economics and green computing.

Cloud Computing

Cloud computing is key in nowadays fast-paced competitive environment, where organizations must be agile and flexible to meet changing market requirements, i.e., rapid expansion and upgrade of resources while meeting stagnant IT budgets. Cloud computing addresses these challenges efficiently enabling individuals or businesses to use IT resources as a service over the network. It provides highly scalable and flexible computing, which enables provisioning of resources on demand. In addition, users can scale up or scale down the demand for computing resources, including storage capacity, with minimal management effort or service provider interaction. Cloud computing empowers self-service requesting through a fully automated request-fulfillment process and consumption-based metering; therefore, consumers pay only for the resources they use, such as CPU hours used, amount of data transferred, and bytes of data stored. Cloud

is usually built upon virtualized data centers, which provide resource pooling and rapid provisioning of resources; occasionally, it could be provided on bare metal servers without the underlying virtualization, which in this case is enclosed within cloud deployment software.

Data Center Fundamentals

Application

Applications are computer programs that provide the logic for computing operations, where they send requests to the underlying operating system to perform read/write (R/W) operations on the storage devices. Applications can be layered on the database, which in turn uses the OS services to perform R/W operations on the storage devices. Applications deployed in a data center environment are commonly categorized as business applications, infrastructure management applications, data protection applications, and security applications. Some examples of these applications are e-mail, enterprise resource planning (ERP), a decision support system (DSS), resource management, backup, authentication and antivirus applications, etc. The characteristics of I/O's (Input/Output) generated by the application influence the overall performance of storage system and storage solution designs; therefore, application developers choose the storage system type based on the frequency of I/O read(s) and write(s).

Application Virtualization

Application virtualization breaks the dependency between the application and the underlying platform (OS and hardware). Application virtualization encapsulates the application and the required OS resources within a virtualized container. This technology provides the ability to deploy applications without making any change to

the underlying OS, file system, or registry of the computing platform on which they are deployed. Because virtualized applications run in an isolated environment, the underlying OS and other applications are protected from potential corruptions. There are many scenarios in which conflicts might arise if multiple applications or multiple versions of the same application are installed on the same computing platform; thus, application virtualization eliminates this conflict by isolating different versions of an application and the associated O/S resources in virtualized containers.

Database Management System

A database is a structured way to store data in logically organized tables that are interrelated. A database helps to optimize the storage and retrieval of data. A DBMS controls the creation, maintenance, and use of a database. The DBMS processes an application's request for data and instructs the operating system to transfer the appropriate data from the storage.

Host/Compute

Users store and retrieve data through applications. The computers on which these applications run are referred to as hosts or computer systems. Hosts can be physical or virtual machines. A compute virtualization software enables creating virtual machines on top of physical compute infrastructure. Examples of physical hosts include desktop computers, servers or a cluster of servers, virtual servers, laptops, and mobile devices. A host consists of CPU, memory, I/O devices, and a collection of software to perform computing operations. This software includes the operating system, file system, logical volume manager, device drivers, and so on. This software can be installed individually or may be part of the operating system.

Operating System and Device Driver

In a traditional computing environment, an operating system controls all the aspects of computing. It works between the application and physical components of a computer/host system. One of the services it provides to the application is data access. The operating system also monitors and responds to user actions and the environment. It organizes and controls hardware components and manages the allocation of hardware resources. It provides basic security for the access and usage of all managed resources. An operating system also performs basic storage management tasks while managing other underlying components, such as the file system, volume manager, and device drivers. In a virtualized computing environment, the virtualization layer works between the operating system and the hardware resources. Here the OS might work differently based on the type of the compute virtualization implemented. In a typical implementation, the OS works as a guest and performs only the activities related to application interaction. In this case, hardware management functions are handled by the virtualization layer. A device driver is a special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a disk drive. A device driver enables the operating system to recognize the device and to access and control devices. Device drivers are hardware-dependent and operating-system-specific.

Memory

Memory has been and continues to be, an expensive component of a host. It determines both the size and number of applications that can run on a host. Memory virtualization enables multiple applications and processes, whose aggregate memory requirement is greater than the available physical memory, to run on a host without impacting each other. Memory virtualization is an operating system feature that virtualizes the physical

Random Access Memory (RAM) of a host. It creates a virtual memory with an address space larger than the physical memory space present in the computer/host system. The virtual memory encompasses the address space of the physical memory and part of the disk storage. The operating system utility that manages the virtual memory is known as the Virtual Memory Manager (VMM). The VMM manages the virtual-to-physical memory mapping and fetches data from the disk storage when a process references a virtual address that points to data at the disk storage. The space used by the VMM on the disk is known as a swap space. A swap space, also known as page file or swap file, is a portion of the disk drive that appears like physical memory to the operating system. In a virtual memory implementation, the memory of a system is divided into contiguous blocks of fixed-size pages. A process known as paging moves inactive physical memory pages onto the swap file and brings them back to the physical memory when required. This enables efficient use of the available physical memory among different applications. The operating system typically moves the least used pages into the swap file so that enough RAM is available for processes that are more active. Access to swap file pages is slower than physical memory pages because swap file pages are allocated on the disk drive which is slower than physical memory.

Logical Volume Manager

In the early days, entire disk drive would be allocated to the file system or other data entity used by the operating system or application. The disadvantage was the lack of flexibility. When a disk drive ran out of space, there was no easy way to extend the file system's size. Also, as the storage capacity of the disk drive increased, allocating the entire disk drive for the file system often resulted in underutilization of storage capacity. The evolution of Logical Volume Managers (LVMs) enabled the dynamic extension of

file system capacity and efficient storage management. LVM is software that runs on the computer/host system and manages logical and physical storage. LVM is an intermediate layer between the file system and the physical disk. It can partition a larger-capacity disk into virtual smaller-capacity volumes in a process called partitioning or aggregate several smaller disks to form a larger virtual volume in a process called concatenation. The LVM provides optimized storage access and simplifies storage resource management. It hides details about the physical disk and the location of data on the disk; in addition, it enables administrators to change the storage allocation even when the application is running.

The basic LVM components are physical volumes, volume groups, and logical volumes. In LVM terminology, each physical disk connected to the host system is a physical volume (PV). A volume group is created by grouping together one or more physical volumes. A unique physical volume identifier (PVID) is assigned to each physical volume when it is initialized for use by the LVM. Physical volumes can be added or removed from a volume group dynamically. They cannot be shared between different volume groups; which means, the entire physical volume becomes part of a volume group. Each physical volume is divided into equal-sized data blocks called physical extents when the volume group is created. Logical volumes (LV) are created within a given volume group. LV can be thought of as a disk partition, whereas the volume group itself can be thought of as a disk. The size of LV is based on a multiple of the physical extents. The LV appears as a physical device to the operating system. LV is made up of noncontiguous physical extents and may span over multiple physical volumes. A file system is created on a logical volume. These LVs are then assigned to the application.

A logical volume can also be mirrored to provide enhanced data availability. Today, logical volume managers are mostly offered as part of the operating system.

File System

A file is a collection of related records or data stored as a unit with a name. A file system is a hierarchical structure of files. A file system enables easy access to data files residing within a disk drive, a disk partition, or a logical volume. A file system consists of logical structures and software routines that control access to files. It provides users with the functionality to create, modify, delete, and access files. Access to files on the disks is controlled by the permissions assigned to the file by the owner, which are also maintained by the file system. In addition, a file system organizes data in a structured hierarchical manner via the use of directories, which are containers for storing pointers to multiple files. All file systems maintain a pointer map to the directories, subdirectories, and files that are part of the file system. If there is no LVM, then there are no logical extents; therefore, without LVM, file system blocks are directly mapped to disk sectors. A file system block is the smallest unit allocated for storing data. Each file system block is a contiguous area on the physical disk. The block size of a file system is fixed at the time of its creation. The file system size depends on the block size and the total number of files system blocks. A file can span multiple file system blocks because most files are larger than the predefined block size of the file system. File system blocks cease to be contiguous and become fragmented when new blocks are added or deleted. Over time, as files grow larger, the file system becomes increasingly fragmented. Apart from the files and directories, the file system also includes a number of other related records, which are collectively called the metadata. The metadata of a file system must be consistent for the file system to be considered healthy. Examples of

some common file systems are FAT 32 (File Allocation Table) and NT File System (NTFS) for Microsoft Windows, UNIX File System (UFS) and Extended File System (EXT2/3) for Linux.

Computer and Desktop Virtualization

Compute virtualization is a technique of masking or abstracting the physical hardware from the operating system. It enables multiple operating systems to run concurrently on a single or clustered physical machine(s). This technique enables creating portable virtual computing systems called virtual machines (VMs). Each VM runs an operating system and application instance in an isolated manner. Compute virtualization is achieved by a virtualization layer that resides between the hardware and virtual machines. This layer is also called the hypervisor. The hypervisor provides hardware resources, such as CPU, memory, and network to all the virtual machines. Within a physical server, a large number of virtual machines can be created depending on the hardware capabilities of the physical server. A virtual machine is a logical entity but appears like a physical host to the operating system, with its own CPU, memory, network controller, and disks. However, all VMs share the same underlying physical hardware in an isolated manner. From a hypervisor perspective, virtual machines are discrete sets of files that include the VM configuration file, data files, and so on. Typically, a physical server often faces resource-conflict issues when two or more applications running on the server have conflicting requirements. For example, applications might need different values in the same registry entry, different versions of the same DLL, and so on. These issues are further compounded with an application's high-availability requirements. As a result, the servers are limited to serve only one application at a time. This causes organizations to purchase new physical machines for

every application they deploy, resulting in expensive and inflexible infrastructure. On the other hand, many applications do not take full advantage of the hardware capabilities available to them. Consequently, resources such as processors, memory, and storage remain underutilized.

Compute virtualization enables to overcome these challenges by allowing multiple operating systems and applications to run on a single physical machine. This technique significantly improves server utilization and provides server consolidation. Server consolidation enables organizations to run their data center with fewer servers. This, in turn, cuts down the cost of new server acquisition, reduces operational cost, and saves data center floor and rack space. Creation of VMs takes less time compared to a physical server setup; organizations can provision servers faster and with ease. Individual VMs can be restarted, upgraded, or even crashed, without affecting the other VMs on the same physical machine. Moreover, VMs can be copied or moved from one physical machine to another without causing application downtime. With the traditional desktop, the OS, applications, and user profiles are all tied to a specific piece of hardware. With legacy desktops, business productivity is impacted greatly when a client device is broken or lost. Desktop virtualization breaks the dependency between the hardware and its OS, applications, user profiles, and settings. This enables the IT staff to change, update, and deploy these elements independently. Desktops hosted at the data center and run on virtual machines, whereas users remotely access these desktops from a variety of client devices, such as a laptop, desktop, and mobile devices, also called thin devices. Application execution and data storage are performed centrally at the data center instead of at the client devices. Because desktops run as virtual machines within an organization’s data center, it mitigates the risk of data leakage and theft. It also helps

to perform centralized backup and simplifies compliance procedures. Virtual desktops are easy to maintain because it is simple to apply patches, deploy new applications and OS, and provision or remove users centrally.

Connectivity

Connectivity refers to the interconnection between hosts or between a host and peripheral devices, such as printers or storage devices. The discussion here focuses only on the connectivity between the host and the storage device. Connectivity and communication between host and storage are enabled using physical components and interface protocols. The physical components of connectivity are the hardware elements that connect the host to storage. Three physical components of connectivity between the host and storage are host interface device, port, and cable. A host interface device or host adapter connects a host to other hosts and storage devices. Examples of host interface devices are Host Bus Adapter (HBA) and Network Interface Card (NIC). HBA is an Application-Specific Integrated Circuit (ASIC) board that performs I/O interface functions between the host and storage, relieving the CPU from additional I/O processing workload. A host typically contains multiple HBAs. A port is a specialized outlet that enables connectivity between the host and external devices. An HBA may contain one or more ports to connect the host to the storage device. Cables connect hosts to internal or external devices using copper or fiber optic media. A protocol enables communication between the host and storage. Protocols are implemented using interface devices (or controllers) at both source and destination. The popular interface protocols used for host to storage communications are Integrated Device Electronics/Advanced Technology Attachment (IDE/ATA), Small Computer System Interface (SCSI), Fiber Channel (FC) and Internet Protocol (IP). IDE/ATA is a popular

interface protocol standard used for connecting storage devices, such as disk drives and CD-ROM drives. This protocol supports parallel transmission and therefore is also known as Parallel ATA (PATA) or simply ATA. IDE/ATA has a variety of standards and names. The Ultra DMA/133 version of ATA supports a throughput of 133 MB per second. In a master-slave configuration, an ATA interface supports two storage devices per connector. However, if the performance of the drive is important, sharing a port between two devices is not recommended. The serial version of this protocol supports single bit serial transmission and is known as Serial ATA (SATA). High performance and low-cost SATA has largely replaced PATA in the newer systems. SATA revision 3.0 provides a data transfer rate up to 6 GB/s.

SCSI has emerged as a preferred connectivity protocol in high-end computers. This protocol supports parallel transmission and offers improved performance, scalability, and compatibility compared to ATA. However, the high cost associated with SCSI limits its popularity among home or personal desktop users. Over the years, SCSI has been enhanced and now includes a wide variety of related technologies and standards. SCSI supports up to 16 devices on a single bus and provides data transfer rates up to 640 MB/s for the Ultra-640 version. Serial Attached SCSI (SAS) is a point-to-point serial protocol that provides an alternative to parallel SCSI. A newer version (SAS 2.0) of serial SCSI supports a data transfer rate up to 6 GB/s. Fiber Channel (FC) is a widely used protocol for high-speed communication to the storage device. The FC interface provides gigabit network speed. It provides a serial data transmission that operates over copper wire and optical fiber. The latest version of the FC interface ‘16FC’ allows transmission of data up to 16 GB/s. IP is a network protocol that has been traditionally used for host-to-host traffic. With the emergence of new technologies, an IP network

has become a viable option for host-to-storage communication. IP offers several advantages in terms of cost and maturity and enables organizations to leverage their existing IP-based network. ISCSI and FCIP protocols are common examples that leverage IP for host-to-storage communication.

On the other hand, data is accessed and stored by applications using the underlying infrastructure. The key components of this infrastructure are the operating system (or file system), connectivity, and storage. The storage device can be internal and (or) external to the host. In either case, the host controller card accesses the storage devices using predefined protocols, such as IDE/ATA, SCSI, or FC. IDE/ATA and SCSI are popularly used in small and personal computing environments for accessing internal storage. FC and iSCSI protocols are used for accessing data from an external storage device or subsystems. External storage devices can be connected to the host directly or through the storage network. When the storage is connected directly to the host, it is referred to as Direct-Attached Storage (DAS). Data can be accessed over a network in one of the following ways; block level, file level, or object level. In general, the application requests data from the file system or operating system by specifying the filename and location. The file system maps the file attributes to the logical block address of the data and sends the request to the storage device. The storage device converts the logical block address (LBA) to a cylinder-head-sector (CHS) address and fetches the data.

In a block-level access, the file system is created on a host, and data is accessed on a network at the block level. In this case, raw disks or logical volumes are assigned to the host for creating the file system. In a file-level access, the file system is created on a

separate file server or at the storage side, and the file-level request is sent over a network. Because data is accessed at the file level, this method has higher overhead, as compared to the data accessed at the block level. Object-level access is an intelligent evolution, whereby data is accessed over a network in terms of self-contained objects with a unique object identifier.

Data Protection

Today's data centers house hundreds of disk drives in their storage infrastructure. Disk drives are inherently susceptible to failures due to mechanical wear and tear and other environmental factors, which could result in data loss. The greater the number of disk drives in a storage array, the greater the probability of a disk failure in the array. For example, consider a storage array of 100 disk drives, each with an average life expectancy of 750,000 hours. The average life expectancy of this collection in the array, therefore, is 750,000/100 or 7,500 hours. This means that a disk drive in this array is likely to fail at least once in 7,500 hours. Redundant Arrays of Inexpensive Disks (RAID) is an enabling technology that leverages multiple drives as part of a set that provides data protection against drive failures. In general, RAID implementations also improve the storage system performance by serving I/O's from multiple disks simultaneously. Modern arrays with flash drives also benefit in terms of protection and performance by using RAID. There are two methods of RAID implementation, hardware, and software. Both have their advantages and disadvantages. Software RAID uses host-based software to provide RAID functions and is implemented at the operating-system level. Also, Software RAID implementations offer cost and simplicity benefits when compared with hardware RAID. In addition, Software RAID affects overall system performance due to additional CPU cycles required to perform RAID

calculations. On the downside, Software RAID does not support all RAID levels and is tied to the host operating system; hence, upgrades to software RAID or to the operating system should be validated for compatibility. This leads to inflexibility in the data-processing environment. On the other hand, in hardware RAID implementations, a specialized hardware controller is implemented either on the host or on the array. Controller card RAID is a host-based hardware RAID implementation in which a specialized RAID controller is installed in the host, and disk drives are connected to it. Manufacturers also integrate RAID controllers on motherboards. A host-based RAID controller is not an efficient solution in a data center environment with a large number of hosts. The external RAID controller is an array-based hardware RAID. It acts as an interface between the host and disks. It presents storage volumes to the host, and the host manages these volumes as physical drives. A RAID array is an enclosure that contains a number of disk drives and supporting hardware to implement RAID. A subset of disks within a RAID array can be grouped to form logical associations called logical arrays, also known as a RAID set or a RAID group.

RAID techniques, i.e., striping, mirroring, and parity, form the basis for defining various RAID levels. These techniques determine the data availability and performance characteristics of a RAID set. Striping is a technique of spreading data across multiple drives, more than one, in order to use the drives in parallel. All the read-write heads work simultaneously, allowing more data to be processed in a shorter time and increasing performance, compared to reading and writing from a single disk. Within each disk in a RAID set, a predefined number of contiguously addressable disk blocks are defined as a strip. The set of aligned strips that spans across all the disks within the RAID set is called a stripe. Strip size, also called stripe depth, describes the number of

blocks in a strip, and is the maximum amount of data that can be written to or read from a single disk in the set, assuming that the accessed data starts at the beginning of the strip. All strips in a stripe have the same number of blocks. Having a smaller strip size means that the data is broken into smaller pieces while spreading across the disks. Stripe size is a multiple of stripe size by the number of data disks in the RAID set. For example, in a five disk striped RAID set with a stripe size of 64KB, the stripe size is 320 KB (64KB x 5). Stripe width refers to the number of data strips in a stripe. Striped RAID does not provide any data protection unless parity or mirroring is used.

Mirroring is a technique whereby the same data is stored on two different disk drives, yielding two copies of the data. If one disk drive failure occurs, the data is intact on the surviving disk drive and the controller continues to service the host's data requests from the surviving disk of a mirrored pair. When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair. This activity is transparent to the host. In addition to providing complete data redundancy, mirroring enables fast recovery from disk failure. However, disk mirroring provides only data protection and is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of the data. Mirroring involves duplication of data, where the amount of storage capacity needed is twice the amount of data being stored. Therefore, mirroring is considered expensive and is preferred for mission-critical applications that cannot afford the risk of any data loss. Mirroring improves read performance because read requests can be serviced by both disks. However, write performance is slightly lower than that in a single disk because each write request manifests as two writes on the disk drives. Mirroring does not deliver the same levels of write performance as a striped RAID.

Parity is a method to protect striped data from disk drive failure without the cost of mirroring. An additional disk drive is added to hold parity, a mathematical construct that allows the re-creation of the missing data. Parity is a redundancy technique that ensures the protection of data without maintaining a full set of duplicate data. Calculation of parity is a function of the RAID controller. Parity information can be stored on separate, dedicated disk drives or distributed across all the drives in a RAID set. The first four disks in the figure, labeled D1 to D4, contain the data. The fifth disk, labeled P, stores the parity information, which, in this case, is the sum of the elements in each row. Now, if one of the data disks fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value. Here, for simplicity, the computation of parity is represented as an arithmetic sum of the data. However, parity calculation is a bitwise XOR operation. Compared to mirroring, parity implementation considerably reduces the cost associated with data protection. Consider an example of a parity RAID configuration with five disks where four disks hold data, and the fifth holds the parity information. In this example, parity requires only 25 percent extra disk space compared to mirroring, which requires 100 percent extra disk space. However, there are some disadvantages of using parity. Parity information is generated from data on the data disk. Therefore, parity is recalculated every time there is a change in data. This recalculation is time-consuming and affects the performance of the RAID array. For parity RAID, the stripe size calculation does not include the parity strip. For example in a five (4 + 1) disk parity RAID set with a strip size of 64 KB, the stripe size will be 256 KB (64 KB x 4). Application performance, data availability requirements, and cost determine the RAID level selection. These RAID

levels are defined on the basis of striping, mirroring, and parity techniques. Some RAID levels use a single technique, whereas others use a combination of techniques.

RAID 0 configuration uses data striping techniques, where data is striped across all the disks within a RAID set. Therefore it utilizes the full storage capacity of a RAID set. To read data, all the strips are put back together by the controller. When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously. RAID 0 is a good option for applications that need high I/O throughput. However, if these applications require high availability during drive failures, RAID 0 does not provide data protection and availability. RAID 1 is based on the mirroring technique. In this RAID configuration, data is mirrored to provide fault tolerance. A RAID 1 set consists of two disk drives and every "write" is written to both disks. The mirroring is transparent to the host. During disk failure, the impact on data recovery in RAID 1 is the least among all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery. RAID 1 is suitable for applications that require high availability and cost is no constraint. Most data centers require data redundancy and performance from their RAID arrays. RAID 1+0 combines the performance benefits of RAID 0 with the redundancy benefits of RAID 1. It uses mirroring and striping techniques and combines their benefits. This RAID type requires an even number of disks, the minimum being four. RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. RAID 1+0 is also called a striped mirror. The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of the data are striped across multiple disk drive pairs in a RAID set. When replacing a failed drive, only the mirror is rebuilt. In other words, the disk array controller uses the surviving drive in the mirrored pair for data recovery and continuous

operation. Data from the surviving disk is copied to the replacement disk. RAID 3 stripes data for performance and uses parity for fault tolerance. Parity information is stored on a dedicated drive so that the data can be reconstructed if a drive fails in a RAID set. For example, in a set of five disks, four are used for data and one for parity. Therefore, the total disk space required is 1.25 times the size of the data disks. RAID 3 always reads and writes complete stripes of data across all disks because the drives operate in parallel. There are no partial writes that update one out of many strips in a stripe. Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance. Data is striped across all disks except the parity disk in the array. Parity information is stored on a dedicated disk so that the data can be rebuilt if a drive fails. Unlike RAID 3, data disks in RAID 4 can be accessed independently so that specific data elements can be read or written on a single disk without reading or writing an entire stripe. RAID 4 provides good read throughput and reasonable write throughput. RAID 5 is a versatile RAID implementation. It is similar to RAID 4 because it uses striping. The drives (strips) are also independently accessible. The difference between RAID 4 and RAID 5 is the parity location. In RAID 4, parity is written to a dedicated drive, creating a write bottleneck for the parity disk. In RAID 5, parity is distributed across all disks to overcome the write bottleneck of a dedicated parity disk. RAID 6 works the same way as RAID 5, except that RAID 6 includes a second parity element to enable survival if two disk failures occur in a RAID set. Therefore, a RAID 6 implementation requires at least four disks. RAID 6 distributes the parity across all the disks. The write penalty in RAID 6 is more than that in RAID 5; therefore, RAID 5 writes perform better than RAID 6. The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.

Raid Level	Min Disks	Avail Storage Capacity (%)	Read Performance	Write Performance	Write Penalty	Protection
1	2	50	Better than Single Disk	Slower than Single Disk (Writes Committed to All Disks)	Moderate	Mirror
1+0	4	50	Good	Good	Moderate	Mirror
3	3	[(n-1)/n]*100	Fair for Random Reads and Good for Sequential Reads	Poor to Fair for Small Random Writes and Fair for Large Sequential Writes	High	Parity (support 1 disk failure)
5	3	[(n-1)/n]*100	Good for Random and Sequential Reads	Fair for Random and Sequential Writes	High	Parity (support 1 disk failure)
6	4	[(n-2)/n]*100	Good for Random and Sequential Reads	Poor to Fair for Random and Sequential Writes	Very High	Parity (support 2 disks failure)

Figure 19 - RAID Comparison

RAID 1+0 performs well for workloads that use small, random, write-intensive I/Os. Some applications that benefit from RAID 1+0 are high transaction rate online transaction processing (OLTP), RDBMS temp space and etc. RAID 3 provides good performance for applications that involve large sequential data access, such as data backup or video streaming. RAID 5 is good for random, read-intensive I/O applications and preferred for messaging, medium-performance media serving, and relational database management system (RDBMS) implementations, in which database administrators (DBAs) optimize data access.

A hot spare refers to a spare drive in a RAID array that temporarily replaces a failed disk drive by taking the identity of the failed disk drive. With the hot spare, one of the following methods of data recovery is performed depending on the RAID implementation; either, if parity RAID is used, the data is rebuilt onto the hot spare from the parity and the data on the surviving disk drives in the RAID set, or if mirroring is used, the data from the surviving mirror is used to copy the data onto the hot spare. When a new disk drive is added to the system, data from the hot spare is copied to it. The hot spare returns to its idle state, ready to replace the next failed drive. Alternatively, the hot spare replaces the failed disk drive permanently. This means that it is no longer a hot spare, and a new hot spare must be configured on the array. A hot spare should be large enough to accommodate data from a failed drive. Some systems implement multiple hot spares to improve data availability. A hot spare can be configured as automatic or user-initiated, which specifies how it will be used in the event of disk failure. In an automatic configuration, when the recoverable error rates for a disk exceed a predetermined threshold, the disk subsystem tries to copy data from the failing disk to the hot spare automatically. If this task is completed before the damaged disk fails, the subsystem switches to the hot spare and marks the failing disk as unusable. Otherwise, it uses parity or the mirrored disk to recover the data. In the case of a user-initiated configuration, the administrator has control of the rebuilding process. For example, the rebuild could occur overnight to prevent any degradation of system performance. However, the system is at risk of data loss if another disk failure occurs.

Intelligent Storage System

Business-critical applications require high levels of performance, availability, security, and scalability. A disk drive is a core element of storage that governs the performance

of any storage system. Some of the older disk-array technologies could not overcome performance constraints due to the limitations of disk drives and their mechanical components. RAID technology made an important contribution to enhancing storage performance and reliability, but disk drives, even with a RAID implementation, could not meet the performance requirements of today's applications. With advancements in technology, a new breed of storage solutions, known as intelligent storage systems, has evolved. These intelligent storage systems are feature-rich RAID arrays that provide highly optimized I/O processing capabilities. These storage systems are configured with a large amount of memory named cache and multiple I/O paths and use sophisticated algorithms to meet the requirements of performance-sensitive applications. These arrays have an operating environment that intelligently and optimally handles the management, allocation, and utilization of storage resources. Support for flash drives and other modern-day technologies, such as virtual storage provisioning and automated storage tiering, which has added a new dimension to storage system performance, scalability, and availability.

An intelligent storage system consists of four key components: front end, cache, back-end, and physical disks. An I/O request received from the host at the front-end port is processed through cache and back-end, to enable storage and retrieval of data from the physical disk. A read request can be serviced directly from the cache if the requested data is found in the cache. In modern intelligent storage systems, front end, cache, and back-end are typically integrated on a single board referred as a storage processor or storage controller.

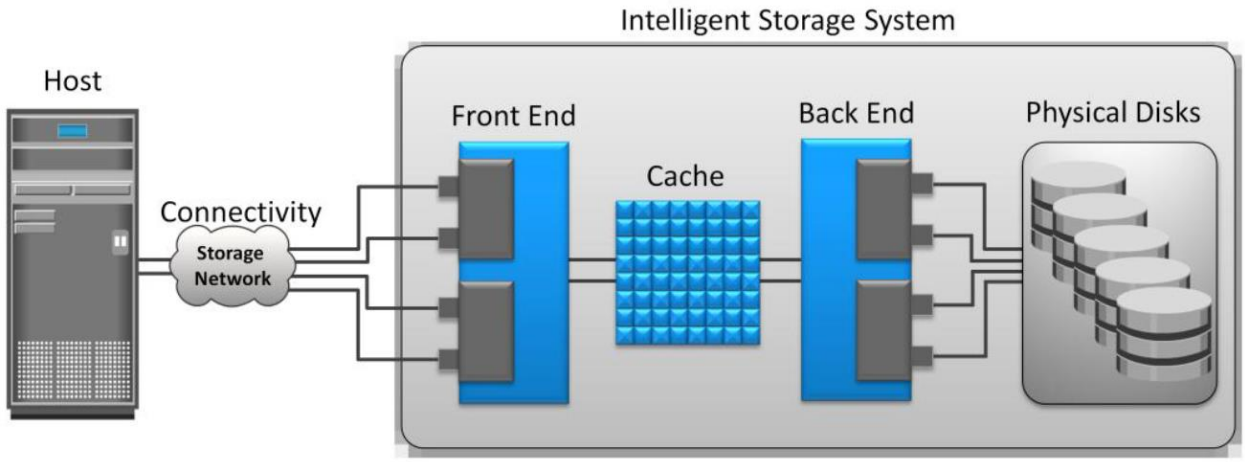

Figure 20 - Intelligent Storage System

The front end provides the interface between the storage system and the host. It consists of two components: front-end ports and front-end controllers. Typically, a front end has redundant controllers for high availability, and each controller contains multiple ports that enable large numbers of hosts to connect to the intelligent storage system. Each front-end controller has processing logic that executes the appropriate transport protocol, such as FibreChannel, iSCSI, FICON, or FCoE for storage connections. Front-end controllers route data to and from cache via the internal data bus. When the cache receives the write data, the controller sends an acknowledgment message back to the host.

Cache is semiconductor memory where data is placed temporarily to reduce the time required to service I/O requests from the host. Cache improves storage system performance by isolating hosts from the mechanical delays associated with rotating disks or hard disk drive (HDD). Rotating disks are the slowest component of an intelligent storage system. Data access on rotating disks usually takes several milliseconds because of “seek” time and rotational latency. Accessing data from the

cache is fast and typically takes less than a millisecond. On intelligent arrays, write data is first placed in the cache and then written to disk.

Server flash-caching technology uses intelligent caching software and PCI Express (PCIe) flash card on the host. This dramatically improves application performance by reducing latency and accelerates throughput. Server flash-caching technology works in both physical and virtual environments and provides performance acceleration for read-intensive workloads. This technology uses minimal CPU and memory resources from the server by offloading flash management onto the PCIe card. It intelligently determines which data would benefit by sitting on the server on PCIe flash and closer to the application. This avoids the latencies associated with I/O access over the network to the storage array. With this, the processing power required for an application's most frequently referenced data is offloaded from the back-end storage to the PCIe card. Therefore, the storage array can allocate greater processing power to other applications.

The back end provides an interface between cache and the physical disks. It consists of two components: back-end ports and back-end controllers. The back-end controls data transfers between the cache and the physical disks. From cache, data is sent to the back end and then routed to the destination disk. Physical disks are connected to ports on the back end. The back-end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage. The algorithms implemented on back-end controllers provide error detection and correction, along with RAID functionality. For high data protection and high availability, storage systems are configured with dual controllers with multiple ports. Such configurations provide an alternative path to physical disks if a controller or port failure occurs. This

reliability is further enhanced if the disks are also dual-ported. In that case, each disk port can connect to a separate controller. Multiple controllers also facilitate load balancing.

Physical disks are connected to the back-end storage controller and provide persistent data storage. Modern intelligent storage systems provide support to a variety of disk drives with different speeds and types, such as FC, SATA, SAS, and flash drives. They also support the use of a mix of flash, FC, or SATA within the same array.

Computing Platforms Evolution

Generally, "platform" term is used refers to data center deployed hardware and software. In addition, computing platforms evolution and growth are directly associated with advancements and changes in technology. The figure below shows the International Data Corporation (IDC) three computing platforms of IT evolution. At a closer look, the first platform, known as Platform 1, goes back to the emergence of computing, where datacenters were primarily based on mainframes and terminals. Consequently, the second platform, known as Platform 2, emerged with the development of the personal computer (PC) in the 1980s, which was later identified by the client-server model, network Ethernet, Database Management Systems, and web applications. Finally, the third platform, known as Platform 3, the focus of today's modern world consists of the four main pillars, i.e., cloud, Big Data, mobile, and social technologies.

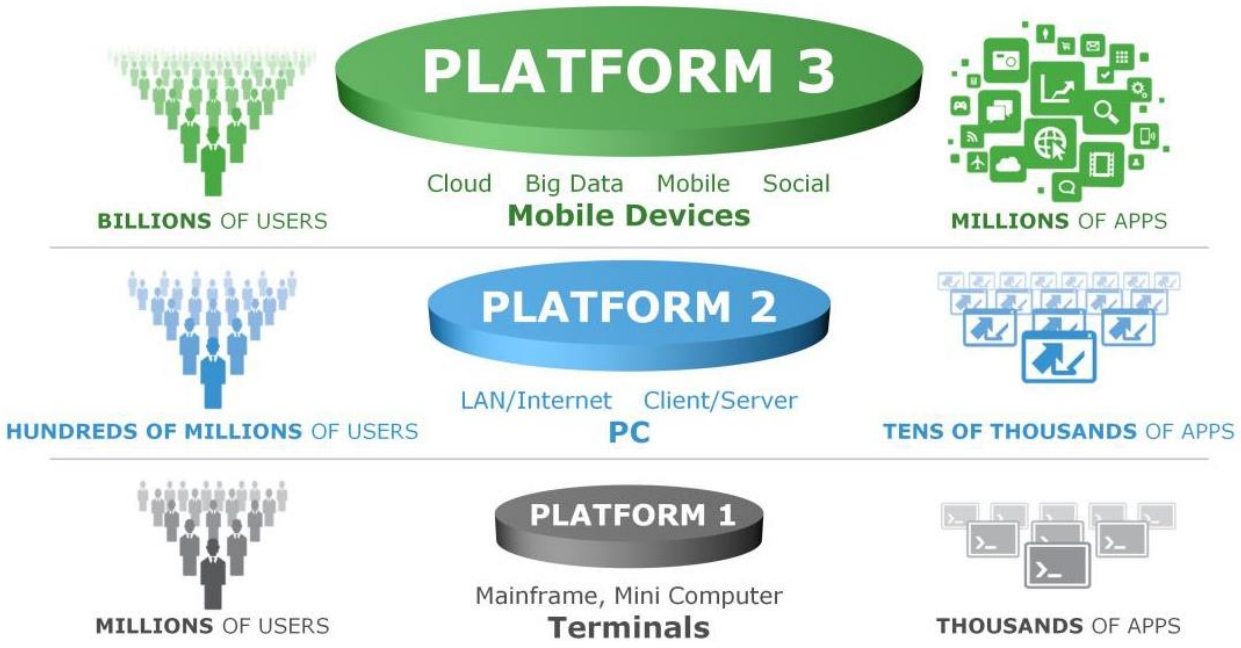

Figure 21 - Computing Platforms Evolution

The computing platforms are usually defined by the encompassing advancements in technologies and innovation; however, this definition is now changing, where computing platforms are defined by the enabling technologies impact in terms of the scale of users and the scope of applications. In more details as per the image above show us, modern world defines computing platforms based on users and applications. Platform 1 used to support millions of users with its associated thousands of applications and solutions, while Platform 2 supported even more with hundreds of millions of users and tens of thousands of applications. Nowadays, the third platform is already supporting billions of users and has contributing millions of applications and software solutions running 24/7. The computing platforms definition by users and apps is in fact is proven when by the stats by IDC showing that over 2.4 billion people, i.e., thirty-six percent of the world's population, are currently connected to the Internet mostly through

smart mobile phone devices, where millions of applications are available on just Windows, iOS, and Android platform devices.

First Platform

The most significant aspect of Platform 1 is the Mainframe, which is a computing system containing very large processing power, memory, and storage capacity. Mainframes were predominantly used in large organizations datacenters to host mission-critical applications and databases. Multiple users simultaneously connect to mainframes through less-powerful devices, such as workstations or terminals. All processing is performed on the mainframe, while the terminals only provide an interface to use the applications and view results.

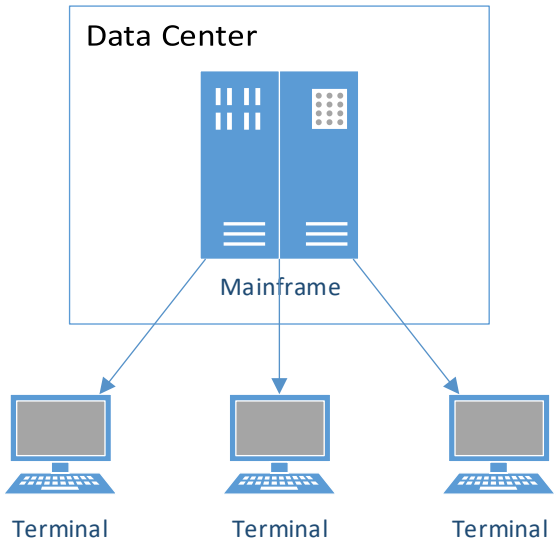

Figure 22 - First Platform

Although mainframes offer high reliability and security, there are several cost concerns associated with them. Mainframes have high acquisition costs, and considerable floor space and energy requirements. Deploying mainframes in a data center may involve substantial capital expense (CAPEX) and operating expense (OPEX). Historically,

large organizations such as banks, insurance agencies, and government departments have used mainframes to run their business operations.

Second Platform

The client-server model uses a distributed application architecture, in which a computing system called “server” runs a program that provides services over a network to other programs running on various end-point devices called “clients”. Server programs receive requests for resources from client programs and in response to the requests, the clients receive access to resources, such as e-mail applications, business applications, web applications, databases, files, and printers. Client devices can be desktops, laptops, and mobile devices. Clients typically communicate with servers over a LAN or WAN, with users making use of either a client application or a web interface on a browser.

In the client-server model, both the clients and the servers may have distinct processing tasks that they routinely perform. For example, a client may run the business application while the server may run the database management system (DBMS) to manage storage and retrieval of information to and from a database. This is called a two-tier architecture. Alternatively, a client may use an application or web interface to accept information while the server runs another application that processes the information and sends the data to a second server that runs the DBMS. This is called the three-tier architecture. This distributed application architecture can be extended to any number of tiers (n-tier architecture). Because both client and server systems are intelligent devices, the client-server model is completely different from the mainframe model.

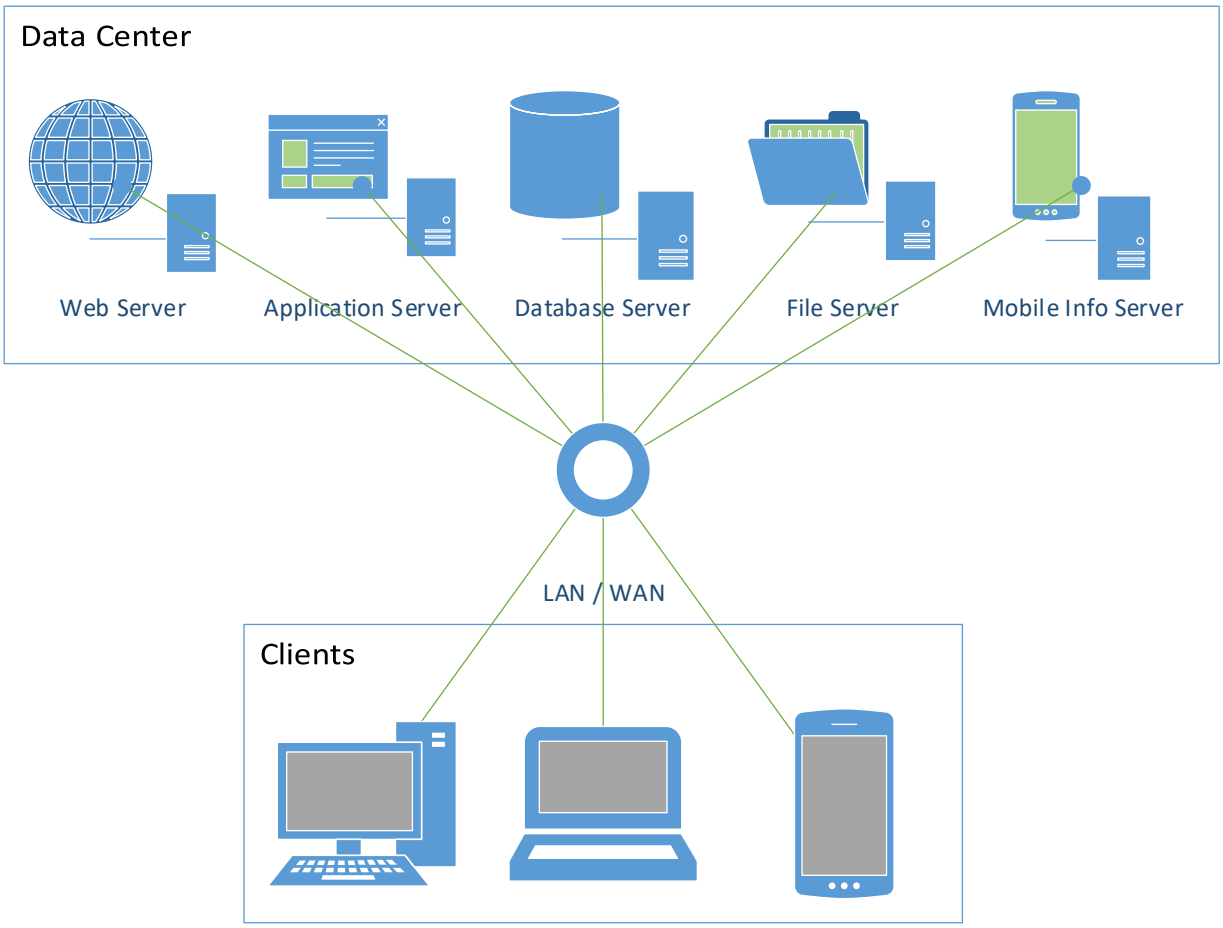

Figure 23 - Second Platform

The figure above shows an example of the client-server model. In the example, clients interact with the web server using a web browser. The web server processes client requests through HTTP and delivers HTML pages. The application server hosts a business application and the database server hosts a DBMS. The clients interact with the application server through client software. The application server communicates with the database server to retrieve information and provide results to the clients. In some implementations, applications and databases may even be hosted on the same server.

Some challenges with the client-server model are associated with the creation of IT silos, maintenance overhead, and scalability issues. In organizations, it is common for business units/departments to have their own servers running business applications. This leads to the creation of application and information silos (individual, disparate systems). Silos make it difficult to efficiently utilize or share IT resources and are challenging to manage and integrate. Though the cost of server hardware is considerably less than mainframes, there is still a significant OPEX involved in the maintenance of multiple servers and clients, and the software running on them. Furthermore, in this model, it is challenging to meet today's rapid growth in users, information, and applications workloads. Adding more servers does not necessarily lead to better workload management. It is also necessary to optimally distribute processing and application logic across servers and application instances.

Third Platform

The term "third platform" was coined by International Data Corporation (IDC), and Gartner refers to the same as a "nexus of forces". The third platform is built on a foundation of cloud, Big Data, mobile, and social technologies. These are the four major "disruptive" technologies that are significantly transforming businesses, economies, and lives globally. At its core, the third platform has the cloud that enables a consumer to provision IT resources as a service from a cloud provider. Big Data enables analytics that create deeper insights from data for improved decision-making. Mobile devices enable pervasive access to applications and information. Social technologies connect individuals and enable collaboration and information exchange.

Over the past three decades, it was essential for organizations to intelligently leverage the second platform for their businesses. According to IDC, over the next three decades, the third platform will represent the basis for solution development and business innovation. The third platform is being used for the digital transformation, evolution, and expansion of all industries and for developing major new sources of competitive advantage. Business strategists, IT leaders, and solution developers are already building disruptive new business models and consumer services around third platform technologies.

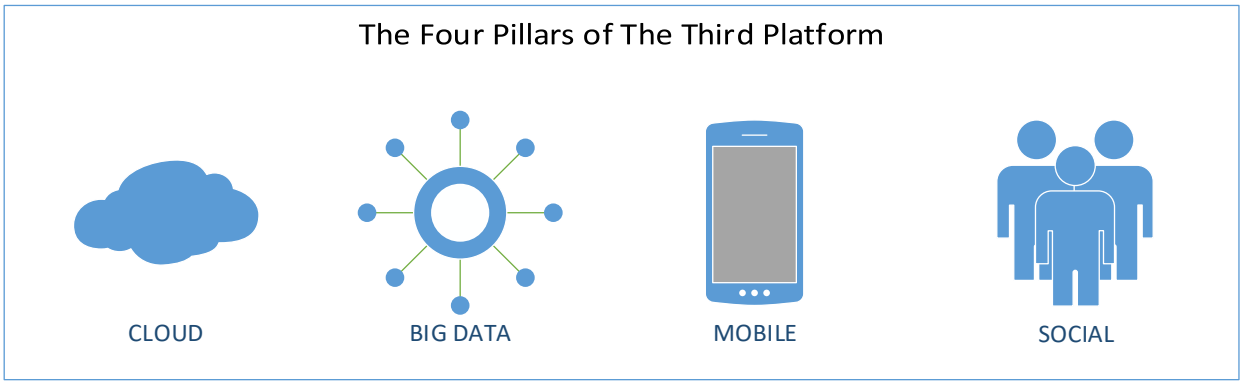

Figure 24 - Third Platform

Third platform technologies are an enhancement of second platform technologies rather than a substitution. A key aspect of the third platform is that it is a convergence of cloud, Big Data, mobile, and social technologies and not just each technology taken in isolation. The real key is combining two or more of the technologies to create high-value industry solutions known as “mashups”. For example, some of the top drivers of the cloud include social and mobile solutions. This means that organizations already see the greatest value in solutions that are mashups across all four technologies. The combinations of third platform technologies are already transforming organizations

such as retail, financial services, government departments, telecommunications, and healthcare.

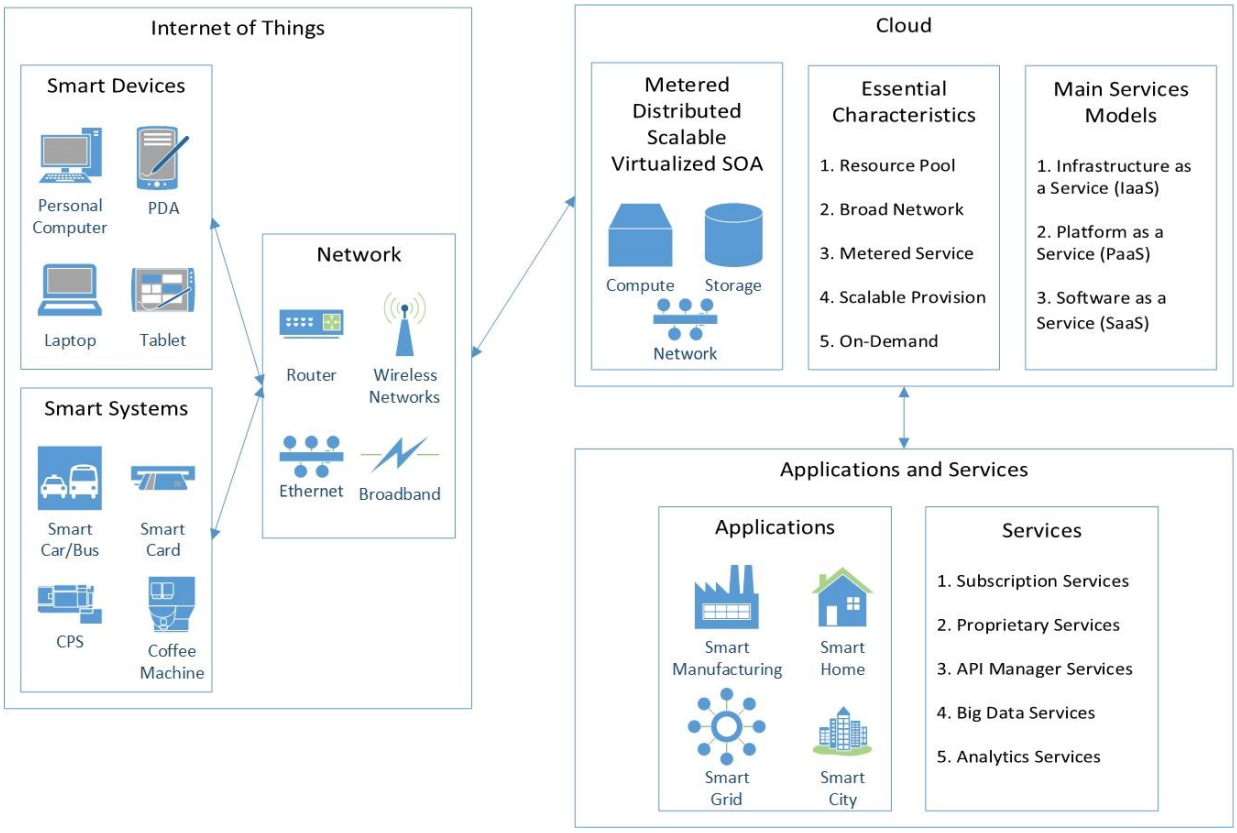

Figure 25 - Third Platform Overview

According to IDC, it is estimated that currently, over 80 percent of the infrastructure and applications in most data centers belong to the second platform. Second platform technologies also currently account for 74 percent of worldwide IT spending. This means that for organizations that have a significant investment in second platform technologies, an immediate and complete shift to the third platform may not be cost-effective and practical. This has led to an intermediate computing platform called "Platform 2.5", between the second and third platforms. Platform 2.5 includes the solutions and technologies that enable organizations to bridge the gap between the second and third platforms. Platform 2.5 technologies enable organizations to use a

combination of second and third platform technologies. Organizations would be able to deliver second platform applications and build third platform outcomes without duplicating and moving data. For example, platform 2.5 technologies would allow an organization to run second platform applications using traditional data structures and protocols while enabling the same data to be leveraged for analytics using Big Data technologies. IDC predicts that future global IT spending will primarily focus on segments such as wireless data, smartphones and tablets, cloud services, Big Data analytics, and IoT. This spending is estimated to be in the hundreds of billions of dollars in each of the segments. This indicates the growing industry trend towards the large-scale adoption of third platform technologies. It is estimated that by 2020 third platform technologies would account for over 40 percent of IT spending.

CHAPTER 4 INDUSTRY 4.0 TECHNOLOGIES DESIGN AND ARCHITECTURE

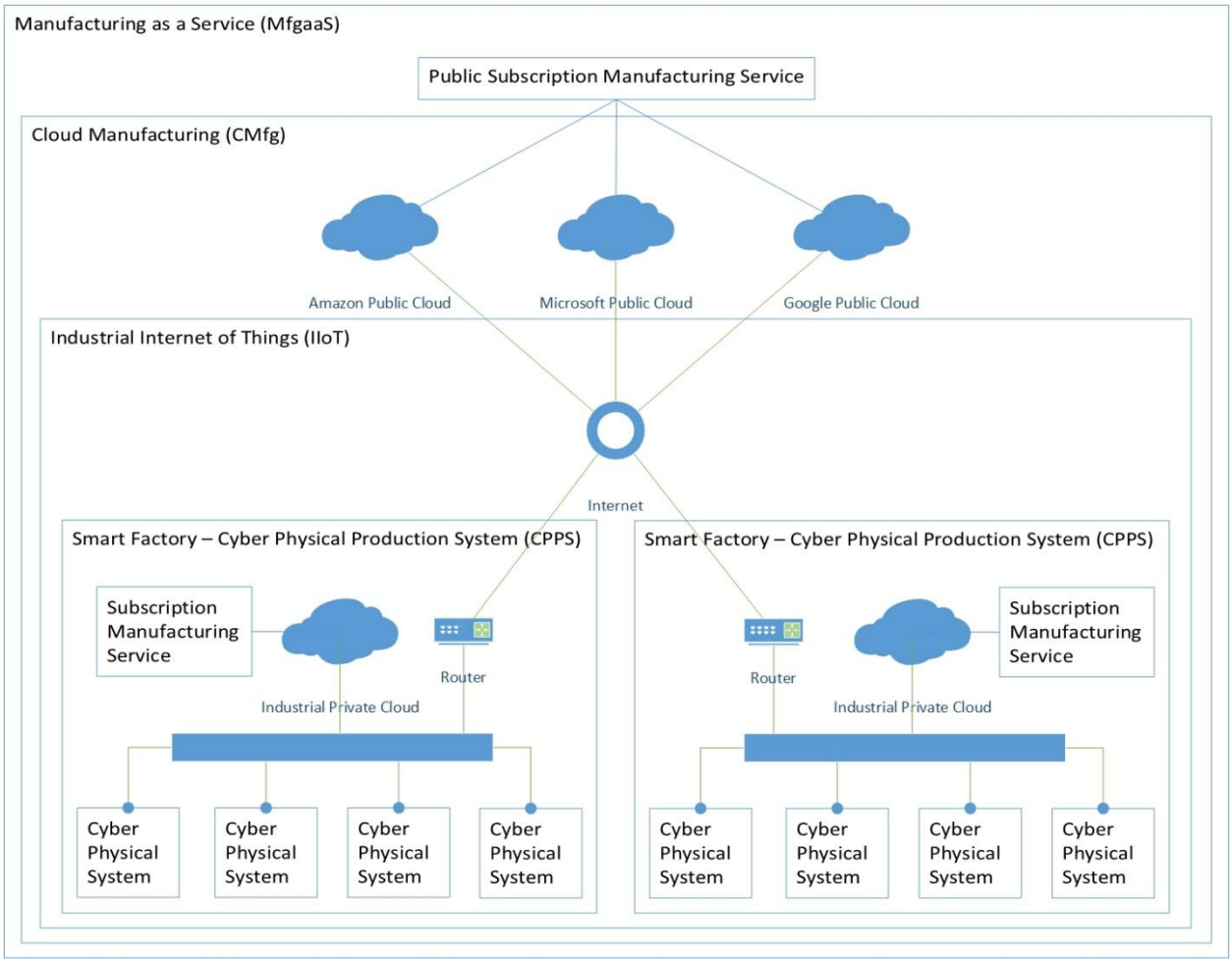

Figure 26 - Industry 4.0 Technologies Overview

Industry 4.0 technologies are CPS, CPPS, IoT, IIoT, CMfg, and MfgaaS as shown above in Figure 26. It all starts with the CPS consisting of sensors, actuators, and many other IoT devices, which are all connected directly to either Intranets or directly to the Internet through wired and/or wireless networks. Ideally, each industrial/manufacturing site will have its own LAN off of which CPSs, Private Cloud, and other management peripherals are connected. Collection of production sites make up CPPSs interconnected with each other using the Internet under the big umbrella of IIoT, where industrial sites along their

computational power extended from their hosted private clouds make up the IIoT. If we further extend the outlook and connect everything to a public cloud for excess computational tasks offloading to the on-demand expandable pay as you go, computational model, we then achieve the CMfg, where many manufacturing/industrial sites are interconnected, sharing, analyzing and processing data, and above all leveraging the cloud computing powers. Not only that, but adding the final layer of applications and their background services running to ensure quality of service, optimization, customization and much more, we reach the end game of MfgaaS, where everyone is a system user from executives, to engineers, to customers, sharing the power of cloud provided manufacturing services with its elite scheduling and order processing techniques.

Applications and Services

Starting from the top tier of applications and services, we deep dive the proposed minimal recommended requirements to have a strong monitoring, reporting, executing, and learning application solution. Figure 27 show our proposed Industry 4.0 application solution deployment mainly consisting of frontends, backends, and connectors.

First, Cyber-physical connectors, also commonly known as device servers or data collectors, act as focal management point(s) to all industrial/manufacturing devices, where all device discovery, data processing in/out, commands/actions, events, (re)configurations, cloud feedback, and much more go through. Connectors are simply virtual machines hosted on the private cloud within the industrial facility and connected through the high-speed WAN/LAN, which in turns connect to the Internet through proper gateways and routers.

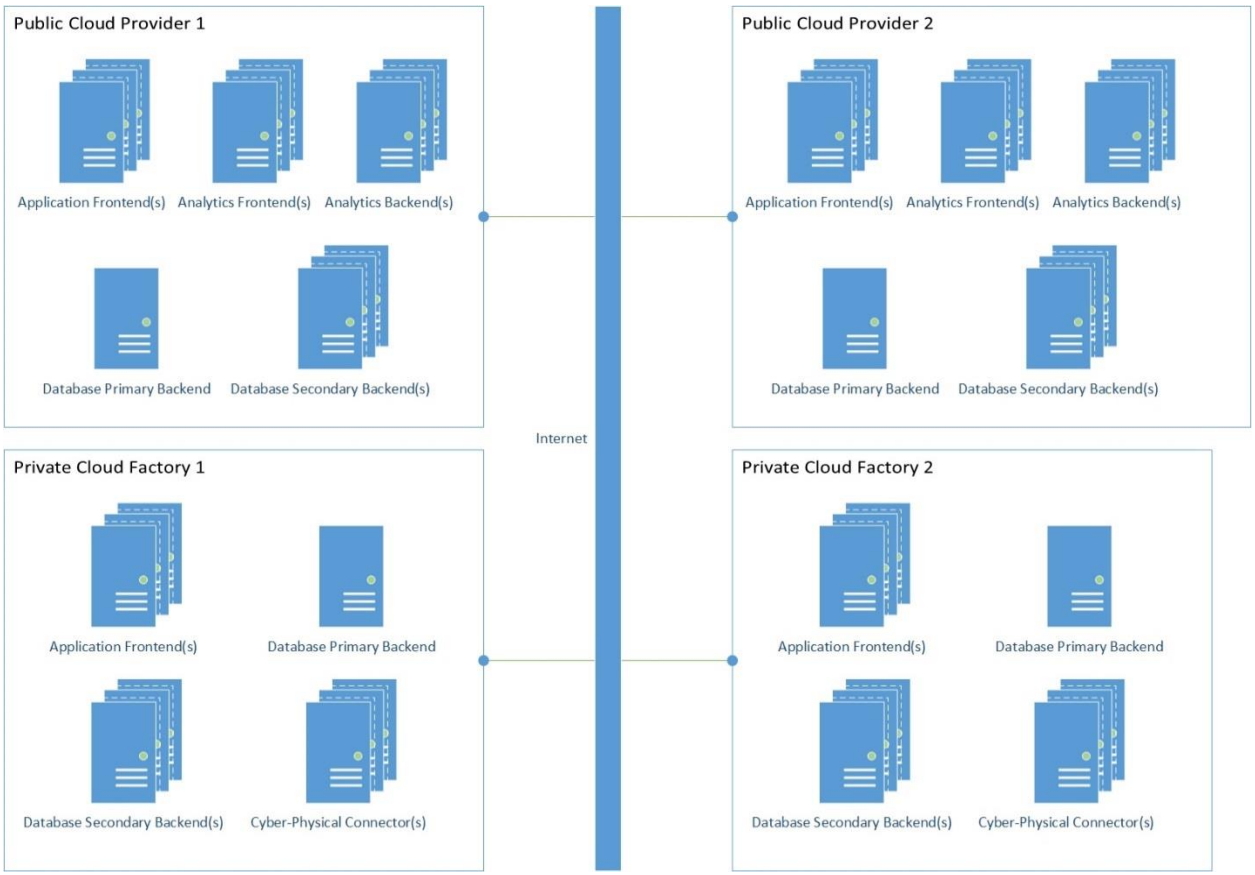

Figure 27 - Industry 4.0 Application Solution Deployment

Second, the Backends responsible for all the heavy processing and storage of data. Backends contain databases and services for alerts, analytics, compliance, events, topology, and security, in addition to web gateway and load balancing. There are two different types of Backends, i.e., Primary and Secondary, depending on the number of devices connected and the frequency of data transactions, actions, and updates. The Backends are responsible for receiving data from any data collection Connector Server, performing validation, and processing data before storing it in the database. As Backends are the only source of data stored in its databases, they require variable huge amounts processing, memory, and storage, which is best suited with Cloud model, i.e., pay as you go and only for what you use offerings.

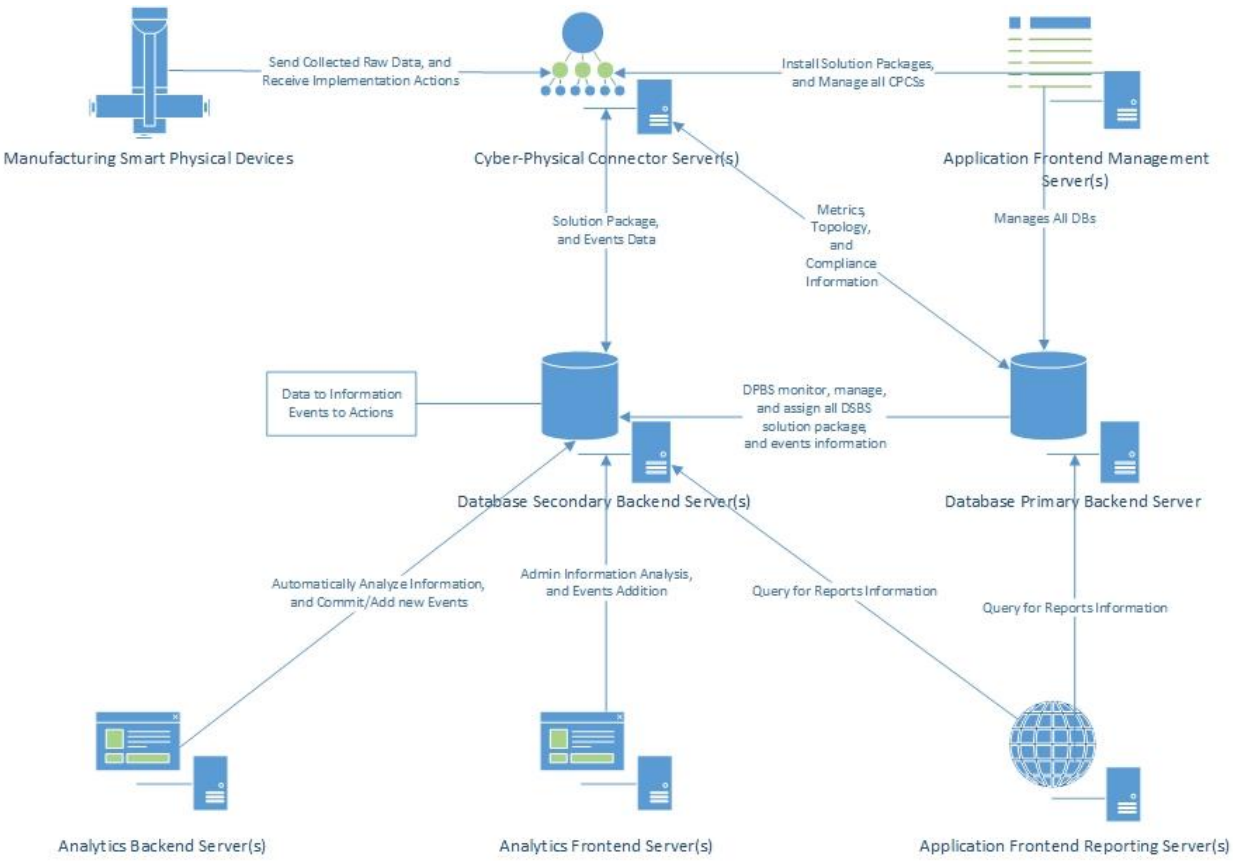

Figure 28 - Industry 4.0 Deployment Overview

Thirdly, the Frontends containing mainly the Web Server Application, e.g., Tomcat, running multiple web, mobile and background applications and services allowing administrators, engineers, DevOps, technicians, managers, executives, vendors, retailers, customers, and any other user type to easily use a convenient User Interface (UI) for both mobile and web on any browsing capable device, e.g., mobile or laptop, for monitoring ongoing operations, administration tasks, development and operations management, report browsing, generation and customization, analytics administration and customization, products viewing, ordering, monitoring and management, and much more. Frontends, as their name indicate, are the portal to viewing/browsing, development, management, monitoring, reporting, and all industrial engineering, vendor and customer related operations. In below Figure 29, we can see a suggested

Industry 4.0 Application and Services Solution Architecture, which shall be explained in further details below.

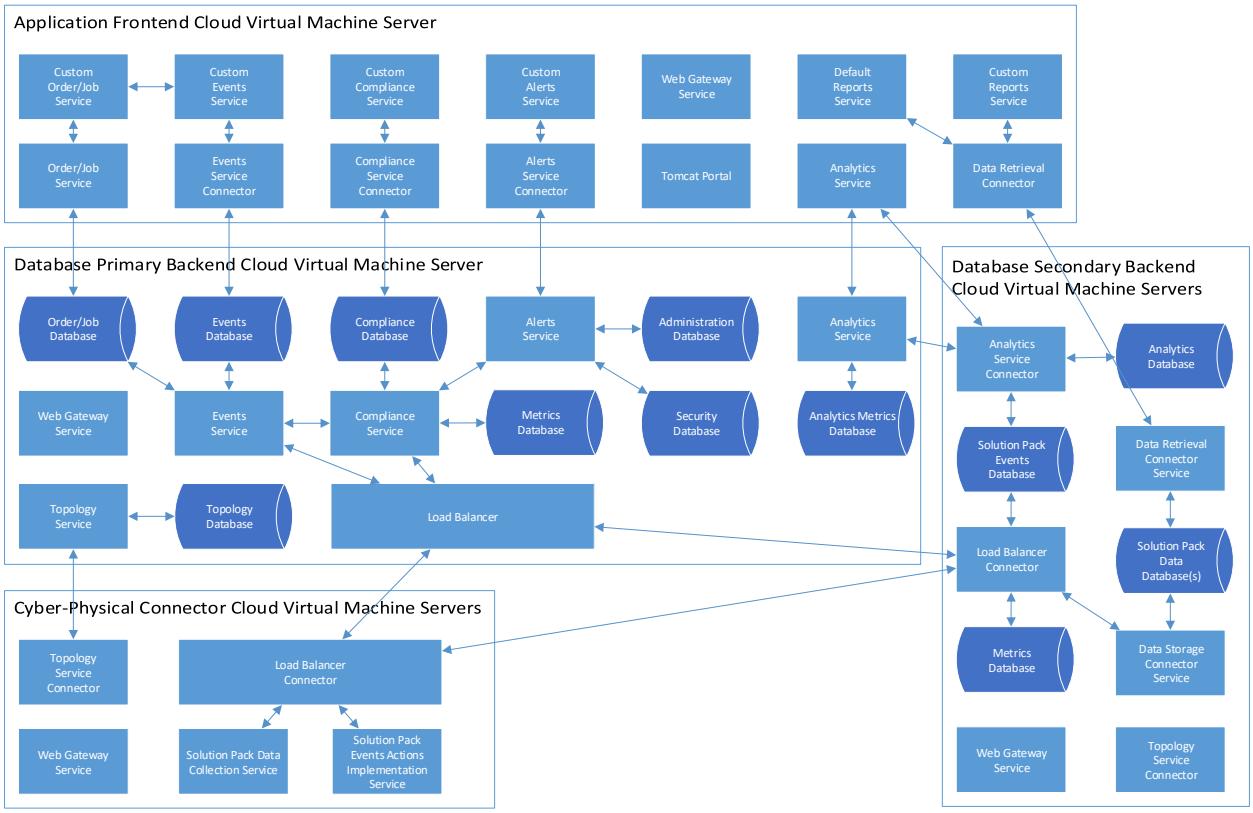

Figure 29 - Industry 4.0 Solution Architecture

Connector

A Cyber-Physical Connector Server (CPCS) is a Virtual Machine operated by either Windows or Linux operating system. Installed on the CPCS are some manufacturing and network messaging and management multiprotocol background brokers/services giving it the ability to communicate with the industrial/manufacturing smart devices, e.g., RabbitMQ, Kafka, MTconnect, etc., where they utilize some well know protocols, e.g.;

- SNMP: Simple Network Management Protocol.
- AMQP: Advanced Message Queuing Protocol.

- MQTT: Message Queue Telemetry Transport.
- STOMP: Simple/Streaming Text Oriented Messaging Protocol.
- HTTP: Hypertext Transfer Protocol.
- XML: Extensible Markup Language.

Managing all these multiprotocol brokers are custom developed data collection and command execution engines. The custom developed engines are named Device Driver Solution Packages (DDSP), where each industrial/manufacturing device has its tailored DDSP.

In addition to DDSP, Connector Servers contain other solution packages, e.g., Web Gateway Service, Load Balancing Connectors, and Topology Service Connectors. The Web Gateway Service Solution Package (WGSSP) is a simple communication interface for forwarding requests between the Industry 4.0 Applications and Services Solution Servers and the Backend Servers. WGSSP is deployed on all Servers deployed in our proposed Solution Architecture. Load Balancing Connectors Solution Packages (LBCSP) are used to load balance the data flow from industrial smart devices to the database Backends. LBCSP are installed on any server sending data to be stored into the Backend database. The Topology Service Connectors Solution Packages (TSCSP) are installed on all Industry 4.0 Solution deployed Servers to maintain the topology and location services necessary to generate visual production line diagrams and track all IoT manufacturing items and their final products.

Backend

The Backend is the backbone of the whole Industry 4.0 Applications and Services Solution. It contains the core components required by the Frontends and Connectors to

perform their basic transactions. Backends are like the engine to a car, where they contain all databases, which are the principal module to all data transactions. They also contain digital signatures of all transactions as an enhanced security measure to be able to trace back any transaction from the point of origin and all of ripple effect reactions. Most data generated from any developed Solutions Pack is stored in the database in a predefined Metrics. All Backends have Metrics Databases and Compliance Services ensuring the authenticity and validity of data before it is stored in mostly the Solutions Packages Data Databases (SPDDB), as they contain the bulk of data or any other database in Industry 4.0 Solution.

Primary Backend

Primary Backend Servers (PBES) are the most loaded servers in terms of the quantity of quality system critical processes and queries they run simultaneously. Firstly and most importantly, PBES contains the main Load Balancer (LB), which is the core engine that controls assigning newly added DDSP to the appropriate Database in one/multiple Secondary Backends that has enough capacity to store all of this DDSP predetermined future data metrics count. Data has expiry date before it is moved from fast accessed storage to the slow access backup storage; this data expiry date affects the overall Metrics count per DDSP, which is set up during installation of DDSP on Connector Servers. Compliance Service Solution Pack (CSSP) works in conjunction with LB to ensure the quality of service of all LB transactions; in addition, CSSP and LB both act as the main pillar for alerts and events services and databases. Smart digital manufacturing Orders are broke down and processed using the Events Service Solution Package (ESSP), which is verified using the CSSP and then passed to the LB to be distributed on each machine according to the event sequencing and smart product line

availability. Alerts Service Solution Package (ASSP) assists this entire process in terms of notifying the LB, CSSP, and ESSP whenever a product has completed one of its order broken down processes/events, accordingly, ready to move to the next production line process. In addition, ASSP is responsible for any manufacturing and production overall and specific system interruptions, where it notifies the subject matter expert engineers and system administrators to take action as soon as the alert is issued. As per Figure 29, each of the Backend Solution Packages has its own database to keep a more detailed log/record of all of its service transactions used later by Data Analytics Control Engineers; in addition, for record keeping, there are the normal device logs kept at each DDSP database during regular log monitoring sequences. Also, PBES contain Administration and Security Databases, which are used to keep a record of all system Users, Profiles, and Roles discussed further in the Frontend section below. PBES hosts the main Topology Service Solution Package (TSSP) made up of both a Service and a Database used to keep track of all production line device and product topology and location to provide a visual aid to system users. Finally, the PBES Big Data Analytics Service Solution Package (BDASSP), similar to other solution packages, devised of a service and database, where the service module is broken down into two main services. BDASSP first service is an off-the-shelf tailored analytics service working as an immediate feedback loop to the digitally controlled subsequent cyber-physical connected devices production line, which differs from the second service working in conjunction with the Frontend Analytics Service administrated by the production line data analytics control engineers monitoring, modifying and tailoring production to reach the optimal performance and quality of production and services.

Secondary Backend

Secondary Backend Servers (SBES) are the main source of DDSP data storage. SBES contain the Data Storage Connector Service Solution Package (DSCSSP), which receives all DDSP data and store them accordingly in the specific DDSP Database (DDSPDB). The DDSPDBs and most other DBs used in the Industry 4.0 Solution Architecture are relational databases, which provide faster deployment and access using the standard Structured Query Language (SQL). SBES also contain Analytics Service Connector Solution Packages (ASCSP) used by the PBESs and Frontend Servers Analytics Services to manage and store all the analytics related data.

Frontend

The Frontend Servers (FES) consists of solution package modules exactly like all servers in our proposed Industry 4.0 Solution Architecture, which means that FES services could be installed in one collective server or distributed among several servers according to the number of users accessing and using each server simultaneously. The ideal server distribution would be that FESs are broken down as such:

- Application Frontend DevOps Server (AFEDOS)
- Application Frontend Production Management Server (AFEPMS)
- Application Frontend Reporting Server (AFERS)
- Application Frontend Big Data Analytics Server (AFBDAS)

AFEDOS is used by the development and operations team, which used to be two separate teams in the past, but due to the rise of the DevOps movement, now we can have a more focused merged development operations team focused on maintaining and administrating the current system, deploying new servers, writing new solution

packages to various production line devices and systems modules, adding new functionalities to enhance UI and general user experience, and generally handling any ICT related development and operation need. AFEPMS used by production line engineers focusing on the industrial part of the business, who can monitor, report, engage and solve existing and forecasted production and manufacturing related issues. AFERS is mainly for executives, managers, and users whom main focus is on generating current production and forecasting reports, monitoring performance, and making/following up on orders related data. Finally, the AFDBAS is as explained in the previous section for Data Analytics Engineers focused on the digital control and overall system improvement, key performance indicators (KPIs) and much more production/business related data analytics topics. On the other hand, if we take a closer look at what Solution Packages shall be installed on the Frontend, we shall see that the FESs are in fact web servers running multiple web, mobile and background applications and services solution packages focused on delivering the best possible support system to the entire Industry 4.0 Solution users ranging from ordering management, regular and custom reporting, events management and customization, compliance monitoring and editing, and, last but not least, big data analytics management, modification, addition, and customization.

Cyber-Physical Production Systems

A Cyber-Physical Production System (CPPS) is the realization of connecting multiple Cyber-Physical Systems (CPS) together under one production roof, then connecting the entire facility to multiple other facilities creating the production systems chain of Industry 4.0 cyber-physical activity.

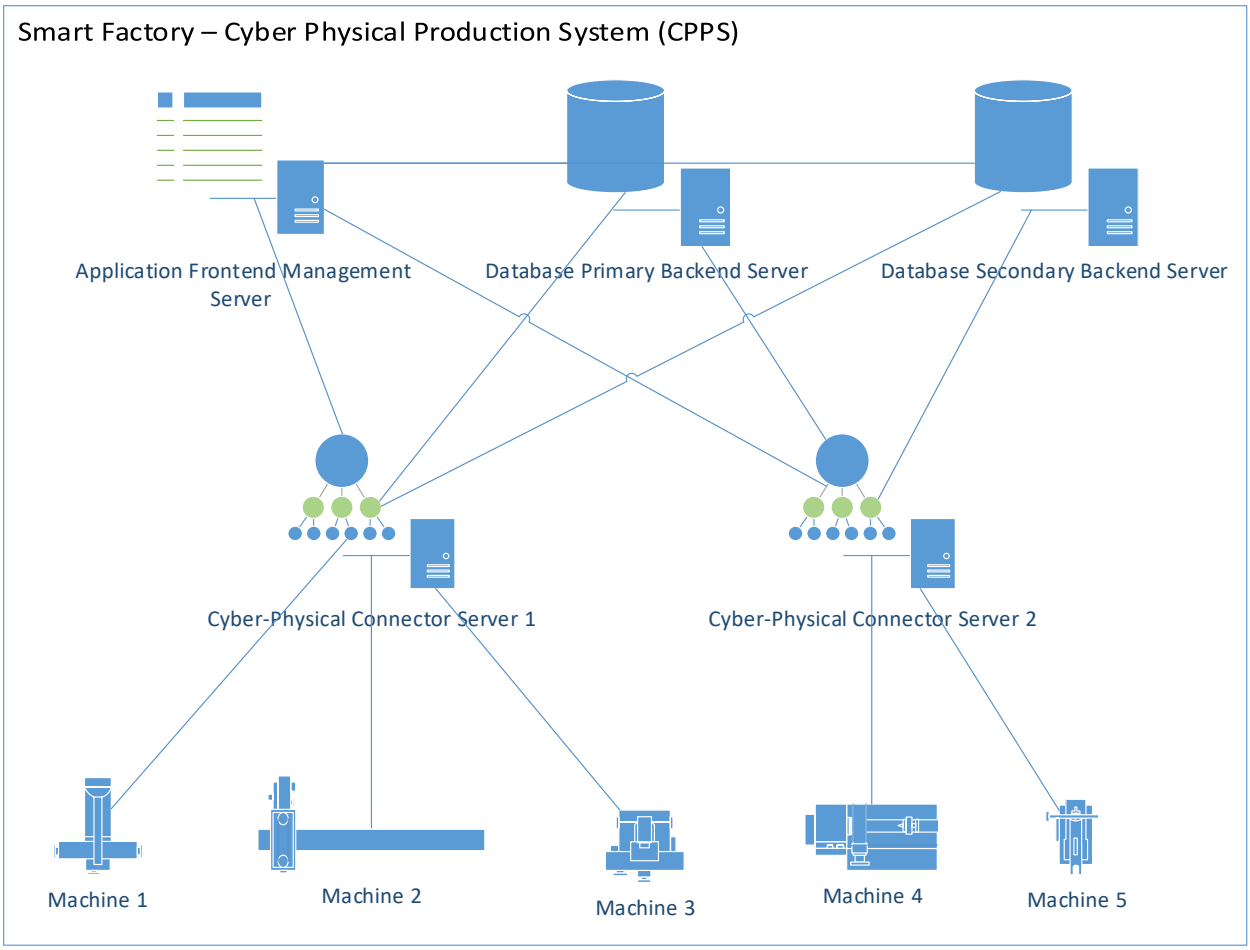

Figure 30 - Industry 4.0 Smart Factory Cyber-Physical Production System

As per our proposed solution architecture, the Industry 4.0 Smart Factory CPPS is the extension of the automation effort of the third industrial revolution, where we have all production line devices switched to become smart devices, i.e., CPSs, capable of advanced digital control functionalities allowing the information and communication technology industry to jump in and enhance the connectivity and performance of all interconnected devices adding a new cyber-digital control signature to the entire production facility.

As per above Figure 30, the CPSs are connected to the CPCS for sensing, monitoring, reporting, and actions, which in term connected to the PBESs, SBESs and FESs

completing the picture of the CPPS Industry 4.0 technological footprint capable of accomplishing much more than just production automation, but, also, customization and performance optimization.

Hybrid Cloud Manufacturing

Cloud Manufacturing (CMfg) is based on both our proposed Industry 4.0 Applications and Services Solution merged with the CPPS creating a new control level from the public cloud. As per previous sections, we have defined how to connect a CPPS using the CPCS, PBES, SBES, and FES. In this section, we show that the technology and solution packages are extended to the Public Cloud and the same modules used in the local cloud are going to be used in the public cloud to deliver CMfg. The question is how to make this work; as per Figure 31 on the following page, we already have a working Industry 4.0 CPPS model, which is deployed on the private/local cloud virtual machines. The key is creating a hybrid cloud of resources, where we pool together multiple heterogeneous data centers cloud implementations into one hybrid cloud allowing access from anywhere and enabling critical applications to remain up and running during any of a variety of planned and unplanned downtime scenarios. Moreover, in a hybrid cloud environment, storage data and virtual machines are migrated from the site to the other allowing for workload rebalancing, disaster avoidance and much smoother data replication and management. An example would be, if the local/private cloud is running out of resources due to the expansion of operations, then using the Hybrid CMfg (HCMfg) we can move/relocate some FESs VMs to the public cloud, which does not require extensive fast access transactions to the CPPS and can adhere to some latency without disrupting operations, e.g., AFERS and AFBDAS.

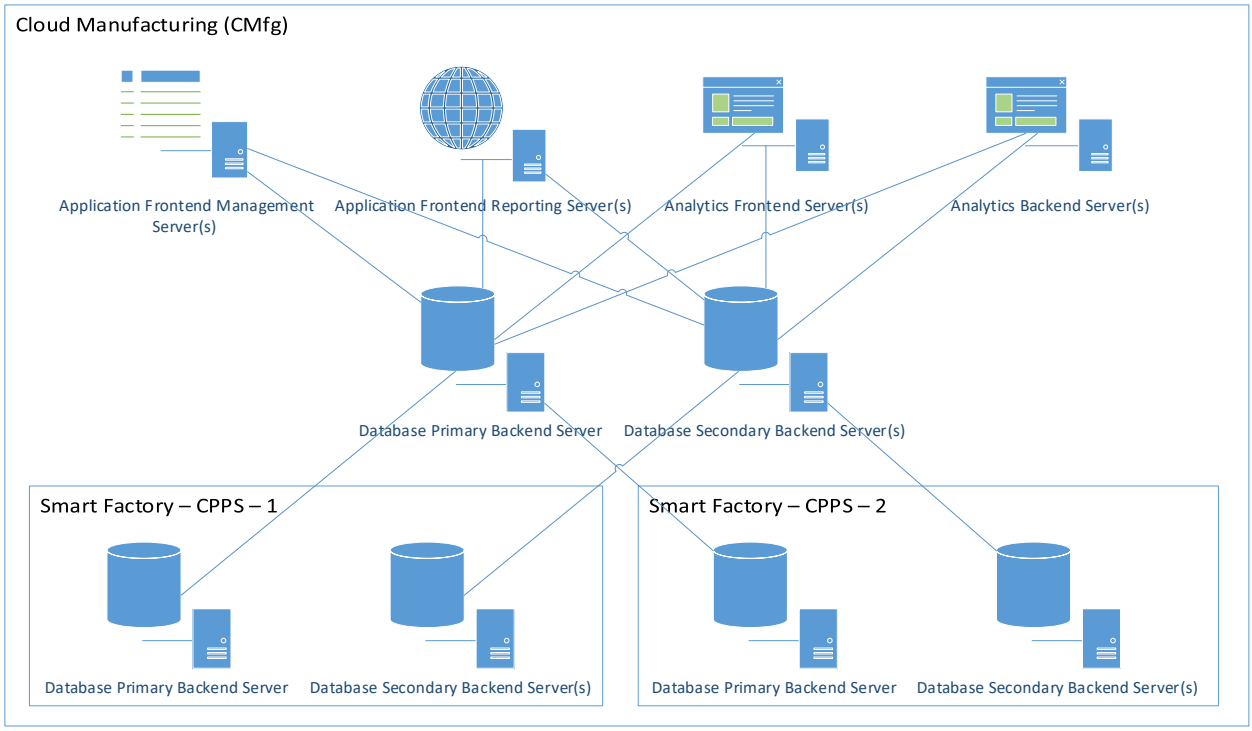

Figure 31 - Industry 4.0 Cloud Manufacturing

Manufacturing as a Service

Manufacturing as a Service (MfgaaS) is the ultimate goal of most big manufacturers out there to have the ability to link all Manufacturing Facilities together with suppliers, retailers, customers and any other user with the capacity to modify, extend, and customize all manufacturing services. This is indeed a remarkable achievement and a true realization of the fourth industrial revolution aimed to involve, enhance and capitalize on customers creating a new level of customer satisfaction and generating new revenue streams.

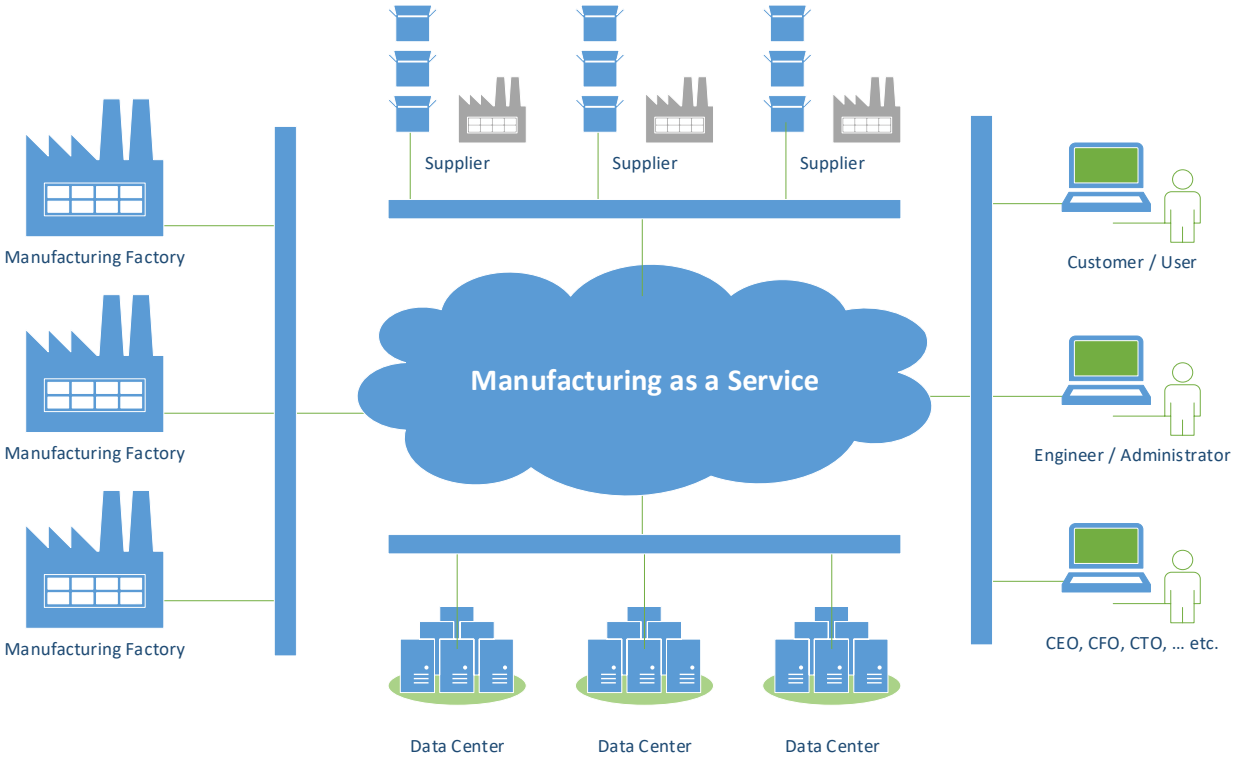

Figure 32 - Industry 4.0 Manufacturing as a Service

The true grasp of Industry 4.0 is only possible using HCMfg, IIoT, CPPS, and ICT applications and services running in the background connecting everyone and everything within a customizable extensible platform of operations. Using the proposed ICT Solution Architecture shall allow Manufacturers to fully utilize their resources and capabilities creating more specialized manufacturing needs and tailoring more Solution Packages by DevOps team(s) to enhance the UI and user experience on all users' levels from customers to executives. To reach this true big accomplishment, it is only possible through strategic partnerships, collaboration, and sharing of all resources and technologies to ensure a better future for manufacturers and to ensure business transformation to meet the new era requirements. After all, there have been so many huge manufacturers throughout the years who lost their businesses due to resistance to change, and not investing in research and development.

CHAPTER 5 GREEN INDUSTRY 4.0

Green Industrial Internet of Things

MECA

The Minimal Energy Consumption Algorithm (MECA) introduced by Huang et al. presents a cost-efficient deployment of green networked IoT. First, the hierarchal green network framework is based on base stations with wired connections to the internet, relay nodes that communicate with each other and back to base stations, and sensor nodes with only directed one-way communication to nearest relay. Second, the optimization model is based on three constraints, which are aimed at contributing to the green network. The constraints are energy consumption, link flow balancing and system budget intended to lower the energy consumption per network nodes, lowering/balancing data flow between nodes and minimize the overall budget. Consequently, the authors introduce the MECA, which tackles an NP-hard problem. Finally, the experimentation is done on a randomly selected IoT node, and it is based on wide-ranging numerical tests, which eventually shows that their proposed model is better than traditional WSN.

Topology and Deployment

IoT network topology consists of base stations, relay nodes, and sensor nodes. The authors start by identifying the topology and deployment schemes for large-scale WSN, which are defined as follows:

- Mesh Topology == Exact Deployment

Mesh is about having sensors nodes acting as relay nodes, which means that each sensor not only captures its own data but also transmits data converging from other nodes. It is the most expensive deployment of them all in terms of energy consumption as it drains the attached batteries and also in terms of cost efficiency as you will end up with nodes having unutilized relay option. Therefore, the lifetime of this deployment is comparatively short to other deployments.

- Plane Topology == Ad-hoc Deployment

Plane or commonly called star topology is deployed where there are only sensor nodes connected to a given base station. This deployment is widely used in the market. However, it is only suitable for small deployments, which makes it not fit for large-scale IoT deployments.

- Hierarchy Topology and Deployment

Hierarchy topology and deployment, also known as a tree topology, is exactly as the name implies nothing but a tiered deployment where sensor nodes can only communicate with a relay or a base station. This deployment method has better routing efficiency than other deployments. However, since it uses a clustering algorithm to determine the hierarchy of the tree, then it is still coming with high complexity on the sensing nodes

- Hybrid Topology == Hierarchy + Ad-hoc Deployment

Hybrid is a powerful topology as it has the same features as the hierarchal topology, but with the exception that sensors nodes can communicate with neighboring sensor nodes. This means that the sensor node is now acting almost like a relay node in the sense that

it can send and receive data. However, with power comes drawbacks, which is simply the more functionality to add to simple sensor nodes the more the complexity of the design, thus the more the cost and energy consumption.

Proposed System Modifications

Huang et al. propose to use the hierarchal topology and deployment, but with modifications that were not yet introduced in any previous work. These modifications are nothing but using clustering, cluster heads, and relay nodes. Typical hierarchal deployments use clustering algorithm in the sensing nodes to form the clustered network and assign relay nodes accordingly to one of the sensing nodes per cluster, which entails an overhead on the sensing node itself as it should have sufficient computational and storage capabilities to execute such complex clustering tasks and be able to act as a relay if it was promoted to cluster head. In comparison, the proposed framework does not require advanced capabilities at the sensing node, thus simplify the node to unidirectional data transfer with much simpler chip design, energy usage, and cost. In traditional hierarchy deployments, relay nodes are chosen by the clustering algorithm per cluster. However, in the authors proposed model, the relay nodes are already predetermined and installed in advance in a mesh formation with the power of routing to each other and to base stations. In similar hierarchal deployments to the proposed one, relay nodes were set to communicate with each other as well, but they were distributed in a tree structure running Breadth First Search (BFS). Cluster heads in known hierarchal topologies are elected or chosen by the clustering algorithm; however, in Huang et al. proposed topology is to have the predetermined relay act as a cluster head. This helps reduce the overhead of using a clustering algorithm on the sensing

node level, which was causing a significant energy loss due to the complexity of the hardware needed to execute such algorithm and to act as a cluster head if needed.

WSN Energy Efficient Strategies

Huang et al. highlighted that energy-saving techniques or strategies in the market are categorized into five main types, i.e., Updating Operating System, Controlling Transmitting Power, Managing Duty Cycle, Routing with Minimized Power, and Clustering for Data Aggregation. Updating the operating system in the sense of dynamically managing system resources in runtime to reduce energy consumption. By optimizing network topology, one can minimize transmit power, which lowers the overall network power consumption. It should not be performed at the expense of network quality nor network lifetime. Allowing unutilized nodes or relays to go to sleep mode to minimize energy consumption. However, it has a great downfall when it comes to trying to discover the sleeping nodes in the network. This tradeoff adds to the complexity of the overall design and network management. The routing with minimized power technique is all about finding the shortest, minimal and best route to transmit data on the network to minimize as much power consumption and energy as possible. Finally, Clustering for Data Aggregation technique is dependent on clustering nodes together and using the cluster head to aggregate data collected from the sensors cluster to save as much energy and transmit power as possible.

Optimization Model Contributions

- Data Energy Consumption

In contrast to traditional techniques that use only transmit power, Huang et al. are going to minimize energy consumption and transmission power by calculating both transmit and receive data collected from all network nodes.

- Data Traffic Balance

To avoid congestion/bottlenecks and delays that can occur at relay nodes as they are the center of communication between the constrained lower layer sensor nodes and the base stations, the authors introduce a link flow balancing constraints.

- Optimization Model

The optimization model is aimed at deploying the sensor nodes in the best efficient way possible to minimize energy consumption and increase the network lifetime.

- Overall System Budget

Huang et al. use a system budget constraint aimed at minimizing the overall cost of the proposed topology framework.

System Framework

Huang et al. argue that majority of IoT devices have a very low mobility percentage, if not fixed, which means that dynamic routing is not suitable for them as it adds to the complexity of the nodes. Thus, they suggest using Static Routing to implement their IoT topology.

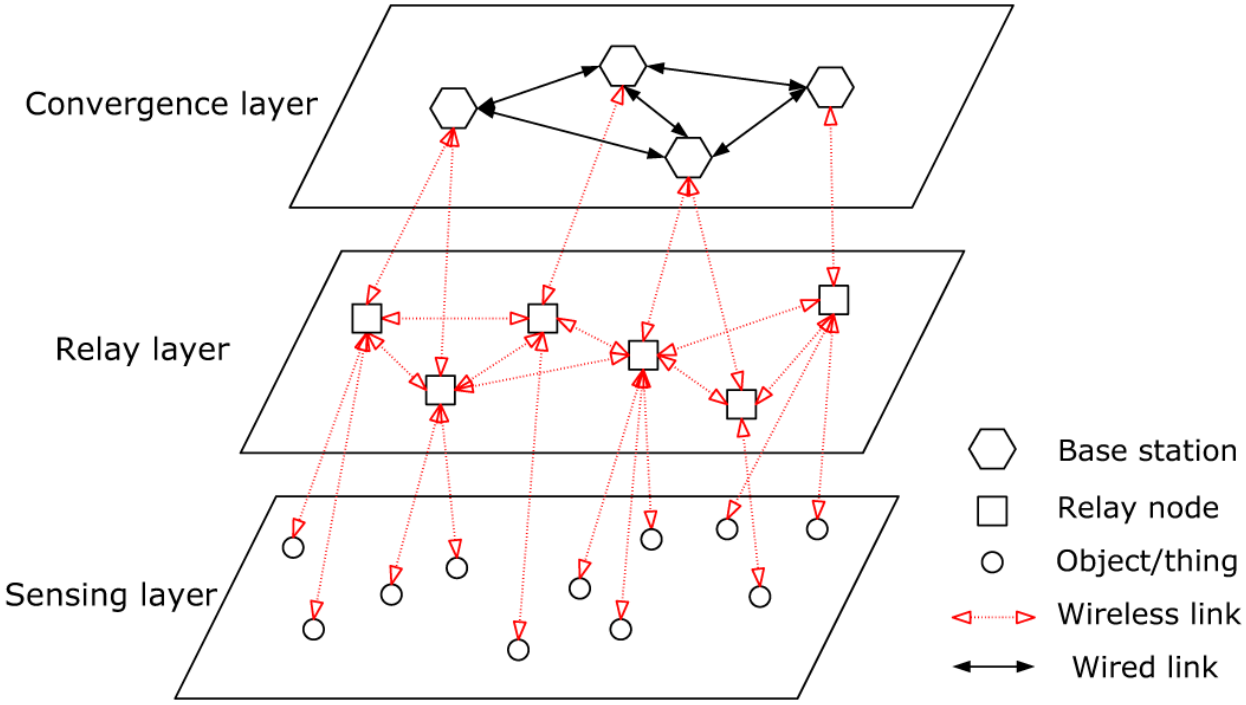

Figure 33 - Green IoT Proposed System Framework (Huang et al.)

Huang et al. introduce how the overall system should look like in the above Figure 33. They specified that the base stations communicate to each other via wired links and to relay nodes via wireless links. Additionally, relay nodes communicate to each other via wireless links and to the sensor nodes, also, via wireless links, but they will ignore in the optimization model the traffic going to sensor nodes as it is insignificant compared to the sensors data collected from the sensor node. Finally, sensor nodes can only communicate to relay nodes and as mentioned earlier the optimization model only consider data transfer from the sensor node to the relay node, but not vice versa.

System Framework Formulation

- Given:

x and y	two points in Euclidean plane
d (x, y)	distance between “x” and “y”

B	set of “n” base stations
R	set of “m” relay nodes
S	set of “l” sensing nodes
r	communication radius of the sensing node
R	communication radius of the relay node
G (N, A)	entire IoT network, where
N	represents the node set
A	represents the link set

- Policy:

For $i \in S$, $j \in S$, i and j cannot communicate with each other, even if $d(i,j) \leq r$

For $i \in S$, $j \in R$, if $d(i,j) < r$, then i can send data to j

For $i \in R$, $j \in (R \cup B)$, if $d(i,j) < R$, then i and j can reach each other

Modeling Green Internet of Things

- Variable Definition:

Etx	Energy consumption at a node for data transmission
Erx	Energy consumption at a node for data receiving
Eelec	Energy consumption of radio electronics
$\epsilon 0$	Transmit amplifier of the node
$\epsilon 1$	Transmit amplifier of the sensing node
$\epsilon 2$	Transmit amplifier of the relay node
dij	The distance between node i and node j
L	Data length
Fij	Data rate from node i and node j
Fmax	Max data rate of a link

CB Monetary cost of a base station

CR Monetary cost of a relay node

CS Monetary cost of a sensing node

l Cardinality of set S

m Cardinality of set R

n Cardinality of set B

W0 System budget

- System Constraints:

G (N, A) is a directed and connected graph. We call node i and node j neighbors, if i and j are able to communicate with each other. Let N(i) be the set of i's neighbors and C be the adjacency matrix of G (N, A), where cij=1 if j∈N(i), otherwise cij=0.

- Energy Consumption Constraint

Data Transmission Energy Consumption: Etx = (Eelec + ϵ0 . d2) . L

Data Receiving Energy Consumption: Erx = Eelec . L

- Consumption of a Sensing Node e_i

$$\boldsymbol{e_i} = \sum_{\boldsymbol{j \in R}} \boldsymbol{c_{ij} \, . \, F_{ij} \, . \, (E^{S}_{elec} + \epsilon_1 \, . \, d^{2}_{ij})} \quad \forall \, \boldsymbol{i \in S}$$

- Consumption of a Relay Node e_j

$$\boldsymbol{e_j} = \sum_{\boldsymbol{i \in S \cup R}} \boldsymbol{c_{ij} \, . \, F_{ij} \, . \, E^{R}_{elec}} + \sum_{\boldsymbol{i \in B \cup R}} \boldsymbol{c_{ji} \, . \, F_{ji} \, (E^{R}_{elec} + \epsilon_2 \, . \, d^{2}_{ji})} \quad \forall \, \boldsymbol{j \in R}$$

- Consumption of a Base Station e_k

$$\boldsymbol{e_k} = \sum_{\boldsymbol{j \in B}} \boldsymbol{c_{jk} \, . \, F_{jk} \, . \, E^{B}_{elec}} \quad \forall \, \boldsymbol{k \in B}$$

, where, E_{elec}^{S} , E_{elec}^{R} , $and\ E_{elec}^{B}$ are the energy consumption of radio electronics of sensor node, relay node and base station.

- Link Flow Balancing Constraint
    - Relay node wireless links should satisfy

$$c_{ij} . F_{ij} + c_{ji} . F_{ji} \leq F_{max} \quad \forall i, j \in R$$

    - Sensor node and base station wireless links should satisfy

$$c_{ij} . F_{ij} \leq F_{max} \quad \forall i \in S, j \in R \ OR \ \forall i \in R, j \in B$$

- System Budget Constraint

$$0 < (C_S . l + C_R . l) < W_0$$

Optimization Model for Green IoT Deployment

$$\min \left[\sum_{i \in S} e_i + \sum_{j \in R} e_j + \sum_{k \in B} e_k \right]$$

s.t.

$$e_i = \sum_{j \in R} c_{ij} . F_{ij} . (E_{elec}^{S} + \epsilon_1 . d_{ij}^2) \quad \forall i \in S$$

$$e_j = \sum_{i \in S \cup R} c_{ij} . F_{ij} . E_{elec}^{R} + \sum_{i \in B \cup R} c_{ji} . F_{ji} (E_{elec}^{R} + \epsilon_2 . d_{ji}^2) \quad \forall j \in R$$

$$e_k = \sum_{j \in B} c_{jk} . F_{jk} . E_{elec}^{B} \quad \forall k \in B$$

$$c_{ij} . F_{ij} + c_{ji} . F_{ji} \leq F_{max} \quad \forall i, j \in R$$

$$c_{ij} . F_{ij} \leq F_{max} \quad \forall i \in S, j \in R \ OR \ \forall i \in R, j \in B$$

$$\mathbf{0} < (\boldsymbol{C_S}.\boldsymbol{l} + \boldsymbol{C_R}.\boldsymbol{l}) < \boldsymbol{W_0}$$

Theorem 1: Above Problem is NP-Hard

Proof: The Key step to resolve the above problem is to map the transmitting/receiving energy for the node pair to a weight on each edge. As such, the problem of finding the minimum energy consumption for the entire system reduces to Steiner Tree problem where the base stations and partial relay nodes (cluster heads) are the destinations, the remainder relay nodes are Steiner points. Since the Steiner tree problem is NP-hard, this problem is NP-hard too.

MECA

- Input:

$$\boldsymbol{S}, \boldsymbol{R}, \boldsymbol{B}, \boldsymbol{R} \geq \boldsymbol{r} > \mathbf{0}$$

- Output:

Minimal Energy Consumption $\mathbf{min}(\boldsymbol{e})$

1: Apply K-means clustering algorithm to obtain a single-cover set $\boldsymbol{S_1} \subseteq \boldsymbol{S}$, choose the closest relay $\boldsymbol{i} \in \boldsymbol{R}$ to replace the $\boldsymbol{j} \in \boldsymbol{S_1}$ forming the set $\boldsymbol{R_1}$.

2: for $\boldsymbol{i} \in \boldsymbol{R}, \boldsymbol{j} \in \boldsymbol{R} \cup \boldsymbol{B}, \boldsymbol{i} \neq \boldsymbol{j}$ do

3: Calculate the distance $\boldsymbol{d_{ij}}$ between $\boldsymbol{i}$ and $\boldsymbol{j}$.

4: if $\boldsymbol{d_{ij}} \leq \boldsymbol{R}$ then

5: Add the node $\boldsymbol{i}$ and $\boldsymbol{j}$ to a candidate set *RN* for placement, set $\boldsymbol{c_{ij}} = \mathbf{1}$ in $\boldsymbol{G}$

6: end if

7: end for

8: Assign edge weight for $\boldsymbol{G}$ in terms of

$$e_i = \sum_{j \in R} c_{ij} \cdot F_{ij} \cdot (E_{elec}^{S} + \epsilon_1 \cdot d_{ij}^2) \quad \forall\, i \in S$$

$$e_j = \sum_{i \in S \cup R} c_{ij} \cdot F_{ij} \cdot E_{elec}^{R} + \sum_{i \in B \cup R} c_{ji} \cdot F_{ji} \, (E_{elec}^{R} + \epsilon_2 \cdot d_{ji}^2) \quad \forall\, j \in R$$

$$e_k = \sum_{j \in B} c_{jk} \cdot F_{jk} \cdot E_{elec}^{B} \quad \forall\, k \in B$$

9: Apply a well-known Steiner Tree algorithm to compute a minimal energy consumption Steiner Tree G^T of $G = (S \cup RN \cup B, A)$ spanning the node set $B \cup R_1$.

10: for each edge in G^T do

11: Sum the total weight on each edge, denote as $\mathbf{min}(e)$

12: end for

13: return $\mathbf{min}(e)$.

- Pseudocode:

[1] Initialize distance R=10 between relays

[2] Initialize number of relays m = [9 * 9] = 81

[3] Generate a fixed distance R=10 matrix R of 81 relay nodes

[4] Generate set S of random l=100 sensor nodes

[5] Generate set R_1 using k-means clustering on set S and finding closest relay in set R

[a] Apply K-Means Clustering to set S.

[b] Plot generated Sensors, Relays and K-Means Clusters.

[c] Generate set R_1 and plot it with Sensor set S.

[6] Generate Dijkstra Tree Topology and Distance of Relay Nodes.

[a] Initializing Topology and Distance Matrices, and Minimal Relay Transmission Distance.

[b] Generate the Topology and Distance Matrices.

[7] Generate and Plot Steiner Tree Elements from Relay Cluster Matrix.

[8] Steiner Tree Algorithm

[a] Initialize Dijkstra and Steiner Tree Elements

[b] Generate and Plot Dijkstra Tree

[c] Generate and Plot Steiner Tree

MECA MATLAB Code

MECA Main Script

```
% ====================================== %
% Mechanical Engineering M.A.Sc. Thesis  %
% Presented to: Prof. Mohamed Elbestawi  %
% Prepared by: Mohamed Aly               %
% ID # 4000-60-328                       %
% Date: Wednesday April 18, 2016         %
% ====================================== %
% IEEE A Novel Deployment Scheme for Green Internet of Things
% Minimal Energy Consumption Algorithm (MECA)
% Main Script


% Clear Workspace and Screen
clear;clc;


% =============== %
% Initializations %
% =============== %
```

```
% Base Station "Fixed"
BaseStation = [50,100];

% Number of Relays
NumberOfRelays = 81;
% Relays: Matrix
RelayMatrix = zeros(NumberOfRelays,2);
% Relays: Matrix Initialization with Relays every 10 Indices "Fixed"
index=1;
for i=10:10:90
   for j=10:10:90
      RelayMatrix(index,1)=i;
      RelayMatrix(index,2)=j;
      index=index+1;
   end
end

% Sensors: Generating Random (x,y) sensor nodes
rng default; % For reproducibility
SensorMatrix = [ randi(100,100,1) , randi(100,100,1) ];

% ================== %
% K-Means Clustering %
% ================== %

% Sensors: Apply K-Means Clustering to Sensor Coordinates
% Note: You can increase/decrease the number of clusters by changing "25"
NumberOfClusters = 25;
```

```
[SensorCluster,SensorClusterCoordinates] = kmeans(SensorMatrix,NumberOfClusters);

% ======================================== %
% Plotting Relays, Sensors & K-Means Clusters %
% ======================================== %

figure('Name','MECA Base Stations, Sensors, Relays & Sensors K-Means Clusters');
plot(BaseStation(:,1),BaseStation(:,2),'go','DisplayName','BaseStation',...
    'MarkerFaceColor',[.49 1 .63],'MarkerSize',10);
hold on;
% Relays: Plotting on figure in Black Squares
plot(RelayMatrix(:,1),RelayMatrix(:,2),'ks','DisplayName','Relays');
% Senosors: Plotting on figure in Red Asterisk
plot(SensorMatrix(:,1),SensorMatrix(:,2),'r*','DisplayName','Sensors');
% Sensors: Plotting Sensor Cluster Coordinates in Blue Diamond
plot(SensorClusterCoordinates(:,1),SensorClusterCoordinates(:,2),'bd','DisplayName','K-Means
Clusters');
legend('show','Location','SouthEast');
title 'MECA Base Stations, Sensors, Relays & Sensors K-Means Clusters';

% ================================================ %
% Generating Relay Set Closest to K-Means Cluster Set %
% ================================================ %

% Relays: Calculate new Relays Matrix based on Clustering
% Note: You need to change the "25" below if you decided to use more/less
% clusters in the above k-means clustering formula
RelayClusterMatrix = zeros(NumberOfClusters,2);
```

```
for i=1:NumberOfClusters
    MinimumDistance = inf;
    for j=1:NumberOfRelays
        position = [ SensorClusterCoordinates(i,:) ; RelayMatrix(j,:) ];
        CalculatedDistance = pdist(position);
        if (CalculatedDistance < MinimumDistance)
            MinimumDistance = CalculatedDistance;
            RelayClusterMatrix(i,:) = RelayMatrix(j,:);
        end
    end
end

% ====================================== %
% Plotting Sensors & Relay Cluster Matrix %
% ====================================== %

figure('Name','MECA Sensors and Clustered Relays');
plot(BaseStation(:,1),BaseStation(:,2),'go','DisplayName','BaseStation',...
    'MarkerFaceColor',[.49 1 .63],'MarkerSize',10);
hold on;
plot(SensorMatrix(:,1),SensorMatrix(:,2),'r*','DisplayName','Sensors');
plot(RelayClusterMatrix(:,1),RelayClusterMatrix(:,2),'ks','DisplayName','Relay Clusters');
legend('show','Location','SouthEast');
title 'MECA Sensors and Clustered Relays';

% ========================================================== %
% Generate Dijkstra Tree Topology and Distance of Relay Nodes %
% ========================================================== %
```

```
% Initializing Topology and Distance Matrices,
% and Minimal Relay Transmission Distance
Topology = zeros(NumberOfRelays+1,NumberOfRelays+1);
Distance = zeros(NumberOfRelays+1,NumberOfRelays+1);
RelayMinimumTransferDistanceR = ceil(pdist([10,10;20,20]));

% Generate the Topology and Distance Matrices
for i=1:NumberOfRelays
   for j=1:NumberOfRelays+1
      if j > NumberOfRelays
         position = [ RelayMatrix(i,:) ; BaseStation(1,:) ];
      else
         position = [ RelayMatrix(i,:) ; RelayMatrix(j,:) ];
      end
      CalculatedDistance = pdist(position);
      if CalculatedDistance < RelayMinimumTransferDistanceR
         Topology(j,i) = 1;
         Distance(j,i) = CalculatedDistance;
      end
   end
end

% ========================================================================
%
% Generating and Plotting Steiner Tree Elements from Relay Cluster Matrix %
% ========================================================================
%
```

```
SteinerTree = zeros(0,(NumberOfClusters+1));
SteinerTreeCoutner = 1;
SteinerTree(SteinerTreeCoutner) = NumberOfRelays+1;
figure('Name','MECA Steiner Tree Initial Elements');
plot(50,100,'o');hold on;
string = sprintf('%g', NumberOfRelays+1);
text(50+0.5, 100+0.1, string);
for node = 1:NumberOfRelays
   x=RelayMatrix(node,1);
   y=RelayMatrix(node,2);
   for i=1:NumberOfClusters
      if ((RelayClusterMatrix(i,1)==x)&&(RelayClusterMatrix(i,2)==y))
         SteinerTreeCoutner = SteinerTreeCoutner + 1;
         SteinerTree(SteinerTreeCoutner) = node;
         plot(x,y,'o');hold on;
         string = sprintf('%g', node);
         text(x+0.5, y+0.1, string);
      end
   end
end
title 'MECA Steiner Tree Initial Elements';

% ======================================== %
% Initializations for Steiner Tree Algorithm %
% ======================================== %

Location = [ RelayMatrix(:,:) ; BaseStation(:,:)];
```

```
Nodes = NumberOfRelays + 1;
% Initializing Steiner Trees Source Elements
SteinerTreesSource=SteinerTree(:,1);
% Getting Number of Rows and Cloumns in specified Steiner Tree
[STrow,STcolumn]=size(SteinerTree);
% For each Steiner Tree
for STindex=1:STrow
    % Initializing Graph
    figure('Name','MECA Dijkstra Tree Initial Topology');
    PlotGraph(Nodes,Location,Topology);

    % Initializing Node related Matrices
    UnVisitedNode=ones(1,Nodes);
    VisitedNode=zeros(1,Nodes);
    PredecessorNode=(inf)*ones(1,Nodes);
    WeightMatrix=(inf)*ones(1,Nodes);

    % ================== %
    % Main Dijkstra Loop %
    % ================== %
    STsource=SteinerTreesSource(STindex);
    PredecessorNode(1,STsource)=STsource;
    WeightMatrix(1,STsource)=0;
    % Add next closest node to the tree
    for Dindex=1:Nodes
        % Find the Closest Node with minimum Weight
        visited = FindClosestNode(Nodes,UnVisitedNode,WeightMatrix);
        % For each Node check if the Node is matching, then set new and old
```

```
    % weights, then check if they are less than the original,
    % If yes, then set the new edge weight, then print
    % the output in the Command Window and plot the new Edge in the
    % Figure initialized above.
    for unvisited=1:Nodes
      if(Topology(visited,unvisited)==1)
        new_weight = WeightMatrix(1,visited) + Distance(visited,unvisited);
        old_weight = WeightMatrix(1,unvisited);
        if ( new_weight < old_weight )
          WeightMatrix(1,unvisited) = new_weight;
          PredecessorNode(1,unvisited) = visited;
        end
      end
    end
    VisitedNode(1,visited)=1;
    UnVisitedNode(1,visited)=0;
    PlotEdge(Location,PredecessorNode,visited);
  end
  title 'MECA Dijkstra Tree Initial Topology';
  % After plotting the Dijkstra Tree in the Figure and Command Window
  % Output the Dijkstra Tree Shortest Paths for Specified Steiner Tree
  for node = 1:Nodes
    fprintf('Tree #%g Dijkstra SHORTEST_PATH (%g, %g) : ',STindex,node,STsource);
    fprintf('%g', node);
    while(node ~= STsource)
      node = PredecessorNode(1,node);
      fprintf(',%g', node);
    end
    fprintf('\n');
```

```
    end
    % Initializing a New Graph to Plot the selected Steiner Tree
    figure('Name','MECA Final Steiner Tree');
    PlotGraph(Nodes,Location,Topology);
    STVisitedCoordinates=[0 0 0 0];
    % Plotting Steiner Tree
    for index = 2:STcolumn
        Flag=0; % Flag to indicate if found edge visited before
        node = SteinerTree(STindex,index);
        fprintf('Tree #%g Steiner SHORTEST_PATH (%g, %g) : ',STindex,node,STsource);
        fprintf('%g', node);
        while(node ~= STsource)
            PlotEdge(Location,PredecessorNode,node);
            node1 = PredecessorNode(1,node);
            x1 = Location(node1,1);
            y1 = Location(node1,2);
            x2 = Location(node,1);
            y2 = Location(node,2);
            [r,c]=size(STVisitedCoordinates);
            for STCIndex=1:r
                if
((x1==STVisitedCoordinates(STCIndex,1))&&(y1==STVisitedCoordinates(STCIndex,2))&&(x2==ST
VisitedCoordinates(STCIndex,3))&&(y2==STVisitedCoordinates(STCIndex,4)))
                    Flag=1;
                end
            end
            if Flag==0
                STVisitedCoordinates = [STVisitedCoordinates ; x1 y1 x2 y2 ];
            end
```

```
            node = PredecessorNode(1,node);
            fprintf(',%g', node);
            Flag=0;
        end
        fprintf('\n');
    end
    title 'MECA Final Steiner Tree';
end
```

Find Closest Node Function

```
% ==================================== %
% Mechanical Engineering M.A.Sc. Thesis  %
% Presented to: Prof. Mohamed Elbestawi  %
% Prepared by: Mohamed Aly               %
% ID # 4000-60-328                       %
% Date: Wednesday April 18, 2016         %
% ==================================== %
% Minimal Energy Consumption Algorithm (MECA)
% Enhanced Minimal Energy Consumption Algorithm (EMECA)
% Both MECA and EMECA Find Closest Node Function


function visited = FindClosestNode(Nodes,UnVisitedNode,WeightMatrix)
visited = inf;
min_weight = inf;
% Loop on all Nodes
for unvisited = 1:Nodes
    % For each unvisited node with Edge Weight less than the minimum
    if (UnVisitedNode(1,unvisited)==1)
        if (WeightMatrix(1,unvisited) < min_weight)
```

```
            min_weight = WeightMatrix(1,unvisited);
            visited = unvisited;
        end
    end
end
end
```

Plot Edge Function

```
% ===================================== %
% Mechanical Engineering M.A.Sc. Thesis  %
% Presented to: Prof. Mohamed Elbestawi  %
% Prepared by: Mohamed Aly               %
% ID # 4000-60-328                       %
% Date: Wednesday April 18, 2016         %
% ===================================== %
% Minimal Energy Consumption Algorithm (MECA)
% Enhanced Minimal Energy Consumption Algorithm (EMECA)
% Both MECA and EMECA Plot Edge Function

function PlotEdge(Location,PredecessorNode,visited)
node1 = PredecessorNode(1,visited);
node2 = visited;
x1 = Location(node1,1);
y1 = Location(node1,2);
x2 = Location(node2,1);
y2 = Location(node2,2);
plot( [x1,x2], [y1,y2], 'r', 'linewidth', 2);
hold on;
end
```

Plot Graph Function

```
% ==================================== %
% Mechanical Engineering M.A.Sc. Thesis  %
% Presented to: Prof. Mohamed Elbestawi  %
% Prepared by: Mohamed Aly               %
% ID # 4000-60-328                       %
% Date: Wednesday April 18, 2016         %
% ==================================== %
% Minimal Energy Consumption Algorithm (MECA)
% Enhanced Minimal Energy Consumption Algorithm (EMECA)
% Both MECA and EMECA Plot Graph Function


function [] = PlotGraph(Nodes,Location,Topology)
% This function allows us to print a graph of given data
% Initializing a Visited Topology Variable to track plotted edges and avoid
% replotting them N*N time.
VisitedTopology=Topology;
for node = 1:Nodes
    x=Location(node,1);
    y=Location(node,2);
    plot(x,y,'o');
    % hold on retains plots in the current axes so that new plots added to
    % the axes do not delete existing plots
    hold on;
    string = sprintf('%g', node);
    text(x+0.5, y+0.1, string);
    for jj = 1:Nodes
        for kk=1:Nodes
```

```
        if ((Topology(jj,kk) == 1) && (VisitedTopology(jj,kk)==1))
            % If we find an unplotted edge (jj,kk)
            % Specify that it is now visited and plot it
            VisitedTopology(jj,kk)=0;
            x1 = Location(jj,1);
            y1 = Location(jj,2);
            x2 = Location(kk,1);
            y2 = Location(kk,2);
            plot( [x1,x2], [y1,y2] );
        end
    end
  end
end
end
```

MECA MATLAB Results

MECA Graphs

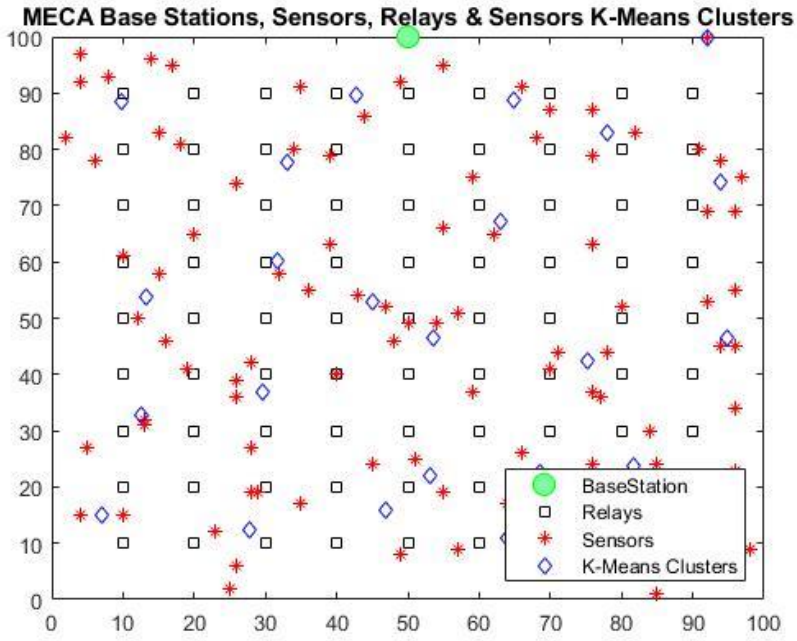

Figure 34 - MECA Algorithm – Base Station, Relays, Sensors and K-Means Clusters

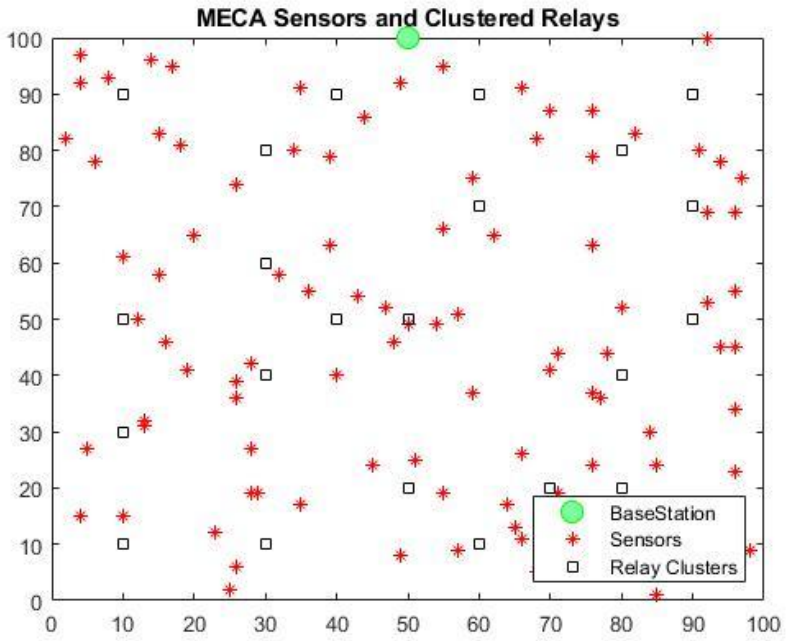

Figure 35 - MECA Algorithm – Base Station, Sensors and Clustered Relays

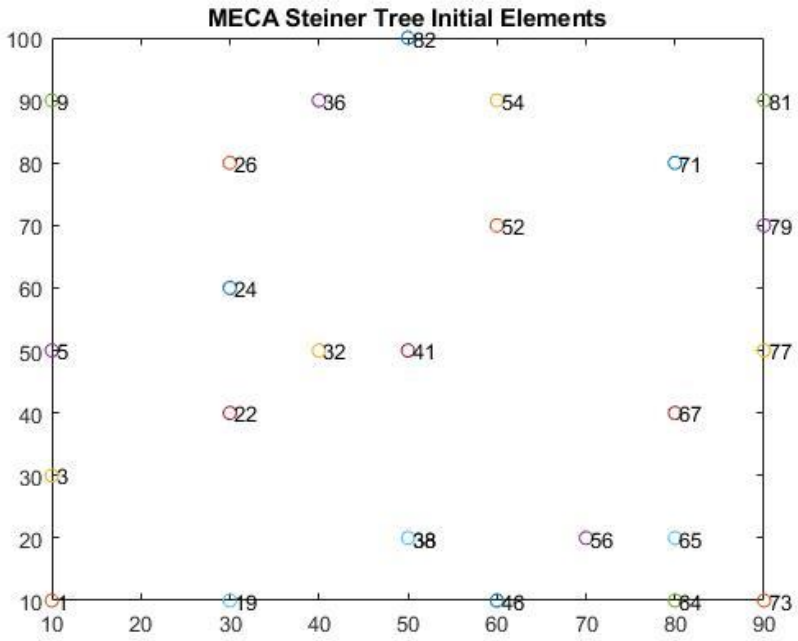

Figure 36 - MECA Algorithm – Steiner Tree Initial Elements

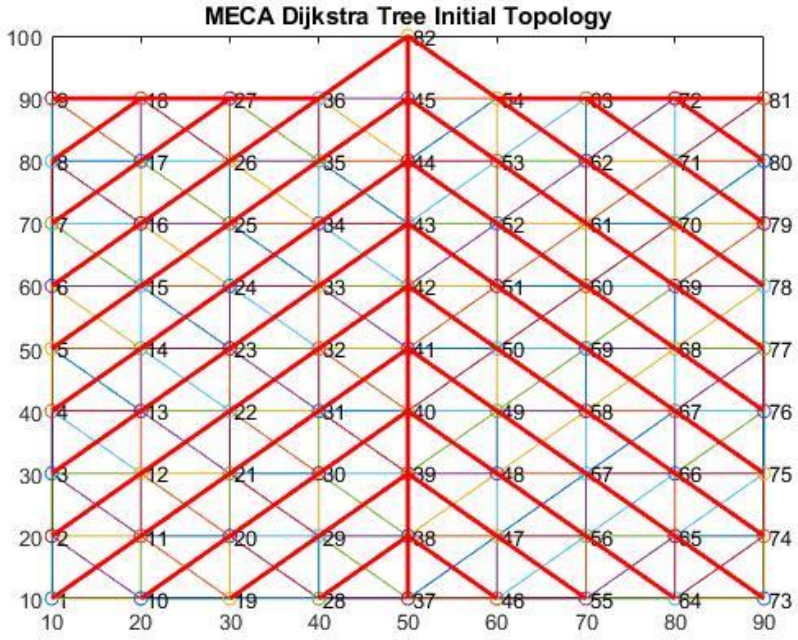

Figure 37 - MECA Algorithm – Dijkstra Tree Initial Topology

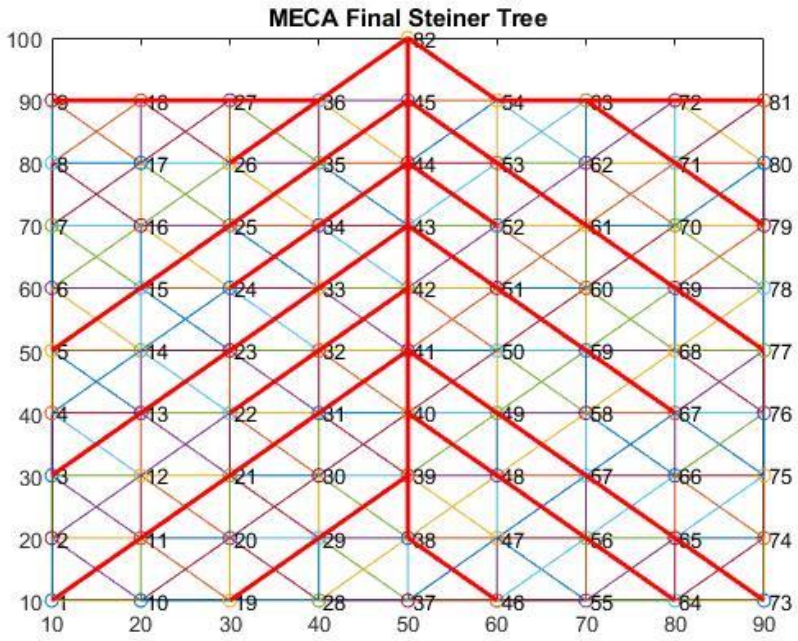

Figure 38 - MECA Algorithm – Final Steiner Tree

MECA MATLAB Command Window Output

Tree #1 Dijkstra SHORTEST_PATH (1, 82) : 1,11,21,31,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (2, 82) : 2,12,22,32,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (3, 82) : 3,13,23,33,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (4, 82) : 4,14,24,34,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (5, 82) : 5,15,25,35,45,82

Tree #1 Dijkstra SHORTEST_PATH (6, 82) : 6,16,26,36,82

Tree #1 Dijkstra SHORTEST_PATH (7, 82) : 7,17,27,36,82

Tree #1 Dijkstra SHORTEST_PATH (8, 82) : 8,18,27,36,82

Tree #1 Dijkstra SHORTEST_PATH (9, 82) : 9,18,27,36,82

Tree #1 Dijkstra SHORTEST_PATH (10, 82) : 10,20,30,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (11, 82) : 11,21,31,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (12, 82) : 12,22,32,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (13, 82) : 13,23,33,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (14, 82) : 14,24,34,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (15, 82) : 15,25,35,45,82

Tree #1 Dijkstra SHORTEST_PATH (16, 82) : 16,26,36,82

Tree #1 Dijkstra SHORTEST_PATH (17, 82) : 17,27,36,82

Tree #1 Dijkstra SHORTEST_PATH (18, 82) : 18,27,36,82

Tree #1 Dijkstra SHORTEST_PATH (19, 82) : 19,29,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (20, 82) : 20,30,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (21, 82) : 21,31,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (22, 82) : 22,32,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (23, 82) : 23,33,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (24, 82) : 24,34,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (25, 82) : 25,35,45,82

Tree #1 Dijkstra SHORTEST_PATH (26, 82) : 26,36,82

Tree #1 Dijkstra SHORTEST_PATH (27, 82) : 27,36,82

Tree #1 Dijkstra SHORTEST_PATH (28, 82) : 28,38,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (29, 82) : 29,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (30, 82) : 30,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (31, 82) : 31,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (32, 82) : 32,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (33, 82) : 33,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (34, 82) : 34,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (35, 82) : 35,45,82

Tree #1 Dijkstra SHORTEST_PATH (36, 82) : 36,82

Tree #1 Dijkstra SHORTEST_PATH (37, 82) : 37,38,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (38, 82) : 38,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (39, 82) : 39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (40, 82) : 40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (41, 82) : 41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (42, 82) : 42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (43, 82) : 43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (44, 82) : 44,45,82

Tree #1 Dijkstra SHORTEST_PATH (45, 82) : 45,82

Tree #1 Dijkstra SHORTEST_PATH (46, 82) : 46,38,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (47, 82) : 47,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (48, 82) : 48,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (49, 82) : 49,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (50, 82) : 50,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (51, 82) : 51,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (52, 82) : 52,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (53, 82) : 53,45,82

Tree #1 Dijkstra SHORTEST_PATH (54, 82) : 54,82

Tree #1 Dijkstra SHORTEST_PATH (55, 82) : 55,47,39,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (56, 82) : 56,48,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (57, 82) : 57,49,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (58, 82) : 58,50,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (59, 82) : 59,51,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (60, 82) : 60,52,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (61, 82) : 61,53,45,82

Tree #1 Dijkstra SHORTEST_PATH (62, 82) : 62,54,82

Tree #1 Dijkstra SHORTEST_PATH (63, 82) : 63,54,82

Tree #1 Dijkstra SHORTEST_PATH (64, 82) : 64,56,48,40,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (65, 82) : 65,57,49,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (66, 82) : 66,58,50,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (67, 82) : 67,59,51,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (68, 82) : 68,60,52,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (69, 82) : 69,61,53,45,82

Tree #1 Dijkstra SHORTEST_PATH (70, 82) : 70,62,54,82

Tree #1 Dijkstra SHORTEST_PATH (71, 82) : 71,63,54,82

Tree #1 Dijkstra SHORTEST_PATH (72, 82) : 72,63,54,82

Tree #1 Dijkstra SHORTEST_PATH (73, 82) : 73,65,57,49,41,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (74, 82) : 74,66,58,50,42,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (75, 82) : 75,67,59,51,43,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (76, 82) : 76,68,60,52,44,45,82

Tree #1 Dijkstra SHORTEST_PATH (77, 82) : 77,69,61,53,45,82

Tree #1 Dijkstra SHORTEST_PATH (78, 82) : 78,70,62,54,82

Tree #1 Dijkstra SHORTEST_PATH (79, 82) : 79,71,63,54,82

Tree #1 Dijkstra SHORTEST_PATH (80, 82) : 80,72,63,54,82

Tree #1 Dijkstra SHORTEST_PATH (81, 82) : 81,72,63,54,82

Tree #1 Dijkstra SHORTEST_PATH (82, 82) : 82

Tree #1 Steiner SHORTEST_PATH (1, 82) : 1,11,21,31,41,42,43,44,45,82

Tree #1 Steiner SHORTEST_PATH (3, 82) : 3,13,23,33,43,44,45,82

Tree #1 Steiner SHORTEST_PATH (5, 82) : 5,15,25,35,45,82

Tree #1 Steiner SHORTEST_PATH (9, 82) : 9,18,27,36,82

Tree #1 Steiner SHORTEST_PATH (19, 82) : 19,29,39,40,41,42,43,44,45,82

Tree #1 Steiner SHORTEST_PATH (22, 82) : 22,32,42,43,44,45,82

Tree #1 Steiner SHORTEST_PATH (24, 82) : 24,34,44,45,82

Tree #1 Steiner SHORTEST_PATH (26, 82) : 26,36,82

Tree #1 Steiner SHORTEST_PATH (32, 82) : 32,42,43,44,45,82

```
Tree #1 Steiner SHORTEST_PATH (36, 82) : 36,82
Tree #1 Steiner SHORTEST_PATH (38, 82) : 38,39,40,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (38, 82) : 38,39,40,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (41, 82) : 41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (46, 82) : 46,38,39,40,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (52, 82) : 52,44,45,82
Tree #1 Steiner SHORTEST_PATH (54, 82) : 54,82
Tree #1 Steiner SHORTEST_PATH (56, 82) : 56,48,40,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (64, 82) : 64,56,48,40,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (65, 82) : 65,57,49,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (67, 82) : 67,59,51,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (71, 82) : 71,63,54,82
Tree #1 Steiner SHORTEST_PATH (73, 82) : 73,65,57,49,41,42,43,44,45,82
Tree #1 Steiner SHORTEST_PATH (77, 82) : 77,69,61,53,45,82
Tree #1 Steiner SHORTEST_PATH (79, 82) : 79,71,63,54,82
Tree #1 Steiner SHORTEST_PATH (81, 82) : 81,72,63,54,82
>>
```

Enhanced MECA (EMECA)

The MECA authors managed to identify a problem in IoT, which is the excessive use of energy based on deploying complex sensing nodes in various IoT topologies. They managed to breakdown the problem into smaller pieces and devise a constraint to solve each individual problem. They introduced an NP-hard optimization problem, for which they came up with an algorithm based on famous Steiner Trees and clustering. Finally, they tested and verified their work. Accordingly, we were able to take their MECA,

implement it using MATLAB, and verify their work using graphs and command window outputs. Additionally, we took spotted areas of improvement in the Minimal Energy Consumption Algorithm and proposed the below Enhanced MECA, which we prove to have less complexity based on the simplified less computing power sensing nodes chipsets, variably set base station based on second round of clustering leading to minimized data link transfer and energy consumption on the hops to and from server, and overall minimized system budget.

- Improvements
    - Replace Base Station and Relays (Wireless Range Extenders/Access Points) with Mesh-Based Wi-Fi System (Router and Satellite Nodes).
    - Improve Optimization Model to replace the fixed Base Station and Replays with variable calculated Mesh-Based Wi-Fi System's Router and Satellite Nodes based on the Sensing double Clustering.
- Justification
    - Having fixed/given Base Station and Relays locations is more of a theoretical approach, rather than a real-life scenario one as the suggested variably selected.
    - In MECA, Relays could be either Wireless Range Extenders (WRE) or Access Points (AP), with the following disadvantages:
        - WRE lose nearly half bandwidth or more from the source.
        - Having multiple WRE connected in Series means tremendous loss in bandwidth and slow connection download and upload speeds.
        - AP requires wired connection to the main source/router.

        - Both WRE and AP create new network SSID, which requires a lot of configurations.
    - Sensors do not need to communicate with each other via relays as this requires the sensors to contain computing capable chipsets requiring more energy, which is not the optimal scenario.

EMECA

- Input:

$\boldsymbol{S, R, B, R \geq r > 0}$

- Output:

1. Apply K-means Clustering Algorithm to obtain a single-cover set SN "Satellite-Nodes" subset of S "Sensors".
2. Apply K-means Clustering Algorithm to obtain a single-cover R "Router".
3. for $\boldsymbol{i \in SN}$ $\boldsymbol{j \in SN \cup R}$, $\boldsymbol{i \neq j}$ do
4. Calculate the distance $\boldsymbol{d_{ij}}$ between $\boldsymbol{i}$ and $\boldsymbol{j}$.
5. if $\boldsymbol{d_{ij} \leq SNR}$ then
    a. Add the node $\boldsymbol{i}$ and $\boldsymbol{j}$ to a candidate set SNN for placement, set $\boldsymbol{c_{ij} = 1}$ in IoT
6. end if
7. end for
8. Assign edge weight for entire IoT network in terms of:
9. $\boldsymbol{e_i = \sum_{j \in SN} c_{ij} \, . \, F_{ij} \, . \, (E_{elec}^{S} + \epsilon_1 \, . \, d_{ij}^2)} \quad \forall \, i \in S$
10. $e_j = \sum_{i \in S \cup SN} c_{ij} \, . \, F_{ij} \, . \, E_{elec}^{SN} + \sum_{i \in R \cup SN} c_{ji} \, . \, F_{ji} \, (E_{elec}^{SN} + \epsilon_2 \, . \, d_{ji}^2) \quad \forall \, j \in SN$
11. $e_k = \sum_{j \in R} c_{jk} \, . \, F_{jk} \, . \, E_{elec}^{R} \quad \forall \, k \in R$

12. Apply Steiner Tree algorithm to compute a IoT^T of $\mathrm{IoT} = (S \cup SNN \cup R, A)$ spanning the node set $\mathrm{R} \cup \mathrm{SN}_1$.
13. for each edge in ToT^T do
    a. Sum the total weight on each edge, denote as $\min(e)$
14. end for
15. return $\min(e)$.

EMECA MATLAB Code

EMECA Main Script

```
% ===================================== %
% Mechanical Engineering M.A.Sc. Thesis  %
% Presented to: Prof. Mohamed Elbestawi  %
% Prepared by: Mohamed Aly               %
% ID # 4000-60-328                       %
% Date: Wednesday April 18, 2016         %
% ===================================== %
% Enhanced Minimal Energy Consumption Algorithm (EMECA)
% Main Script


% Clear Workspace and Screen
clear;clc;


% Sensors: Generating Random (x,y) sensor nodes
rng default; % For reproducibility
SensorMatrix = [ randi(100,100,1) , randi(100,100,1) ];


% ================= %
```

```
% K-Means Clustering %
% ================== %

% Sensors: Apply K-Means Clustering to obtain the Satellite Nodes
% Note: You can increase/decrease the number of clusters by changing "25"
NumberOfClusters = 25;
[Sensors,SatelliteNodesMatrix] = kmeans(SensorMatrix,NumberOfClusters);

% Sensors: Apply K-Means Clustering to obtain the Router location
NumberOfRouters = 1;
[Routers,RouterMatrix] = kmeans(SatelliteNodesMatrix,NumberOfRouters);

% ========================================== %
% Plotting Relays, Sensors & K-Means Clusters %
% ========================================== %

figure('Name','MECA Base Stations, Sensors, Relays & Sensors K-Means Clusters');
plot(RouterMatrix(:,1),RouterMatrix(:,2),'go','DisplayName','Router',...
    'MarkerFaceColor',[.49 1 .63],'MarkerSize',10);
hold on;
% Senosors: Plotting on figure in Red Asterisk
plot(SensorMatrix(:,1),SensorMatrix(:,2),'r*','DisplayName','Sensors');
% Sensors: Plotting Sensor Cluster Coordinates in Blue Diamond
plot(SatelliteNodesMatrix(:,1),SatelliteNodesMatrix(:,2),'ks','DisplayName','Satellite Nodes');
legend('show','Location','SouthEast');
title 'Sensors, Router & Satellite Nodes';

% ======================================================== %
```

```
% Generate Dijkstra Tree Topology and Distance of Relay Nodes %
% ========================================================== %

% Initializing Topology and Distance Matrices,
% and Minimal Relay Transmission Distance
Topology = zeros(NumberOfClusters+1,NumberOfClusters+1);
Distance = zeros(NumberOfClusters+1,NumberOfClusters+1);
SatelliteNodesMinimumTransferDistanceR = ceil(pdist([40,40;20,20]));

% Generate the Topology and Distance Matrices
for i=1:NumberOfClusters
   for j=1:NumberOfClusters+1
      if j > NumberOfClusters
         position = [ SatelliteNodesMatrix(i,:) ; RouterMatrix(1,:) ];
      else
         position = [ SatelliteNodesMatrix(i,:) ; SatelliteNodesMatrix(j,:) ];
      end
      CalculatedDistance = pdist(position);
      if CalculatedDistance < SatelliteNodesMinimumTransferDistanceR
         Topology(j,i) = 1;
         Distance(j,i) = CalculatedDistance;
      end
   end
end

% ====================================================================
%
% Generating and Plotting Steiner Tree Elements from Relay Cluster Matrix %
```

```
% ========================================================================
%


SteinerTree = zeros(0,(NumberOfClusters + 1));
SteinerTreeCoutner = 1;
SteinerTree(SteinerTreeCoutner) = NumberOfClusters + 1;
figure('Name','EMECA Steiner Tree Initial Elements');
plot(RouterMatrix(:,1),RouterMatrix(:,2),'o');hold on;
string = sprintf('%g', NumberOfClusters + 1);
text(RouterMatrix(:,1)+0.5, RouterMatrix(:,2)+0.1, string);
for node = 1:NumberOfClusters
    x=SatelliteNodesMatrix(node,1);
    y=SatelliteNodesMatrix(node,2);
    for i=1:NumberOfClusters
        if ((SatelliteNodesMatrix(i,1)==x)&&(SatelliteNodesMatrix(i,2)==y))
            SteinerTreeCoutner = SteinerTreeCoutner + 1;
            SteinerTree(SteinerTreeCoutner) = node;
            plot(x,y,'o');hold on;
            string = sprintf('%g', node);
            text(x+0.5, y+0.1, string);
        end
    end
end
title 'EMECA Steiner Tree Initial Elements';


% ======================================== %
% Initializations for Steiner Tree Algorithm %
% ======================================== %
```

```
Location = [ SatelliteNodesMatrix(:,:) ; RouterMatrix(:,:)];
Nodes = NumberOfClusters + 1;
% Initializing Steiner Trees Source Elements
SteinerTreesSource=SteinerTree(:,1);
% Getting Number of Rows and Cloumns in specified Steiner Tree
[STrow,STcolumn]=size(SteinerTree);
% For each Steiner Tree
for STindex=1:STrow
    % Initializing Graph
    figure('Name','EMECA Dijkstra Tree Initial Topology');
    PlotGraph(Nodes,Location,Topology);

    % Initializing Node related Matrices
    UnVisitedNode=ones(1,Nodes);
    VisitedNode=zeros(1,Nodes);
    PredecessorNode=(inf)*ones(1,Nodes);
    WeightMatrix=(inf)*ones(1,Nodes);

    % ================== %
    % Main Dijkstra Loop %
    % ================== %
    STsource=SteinerTreesSource(STindex);
    PredecessorNode(1,STsource)=STsource;
    WeightMatrix(1,STsource)=0;
    % Add next closest node to the tree
    for Dindex=1:Nodes
        % Find the Closest Node with minimum Weight
```

```
    visited = FindClosestNode(Nodes,UnVisitedNode,WeightMatrix);
    % For each Node check if the Node is matching, then set new and old
    % weights, then check if they are less than the original,
    % If yes, then set the new edge weight, then print
    % the output in the Command Window and plot the new Edge in the
    % Figure initialized above.
    for unvisited=1:Nodes
      if(Topology(visited,unvisited)==1)
        new_weight = WeightMatrix(1,visited) + Distance(visited,unvisited);
        old_weight = WeightMatrix(1,unvisited);
        if ( new_weight < old_weight )
          WeightMatrix(1,unvisited) = new_weight;
          PredecessorNode(1,unvisited) = visited;
        end
      end
    end
    VisitedNode(1,visited)=1;
    UnVisitedNode(1,visited)=0;
    PlotEdge(Location,PredecessorNode,visited);
  end
  title 'MECA Dijkstra Tree Initial Topology';
  % After plotting the Dijkstra Tree in the Figure and Command Window
  % Output the Dijkstra Tree Shortest Paths for Specified Steiner Tree
  for node = 1:Nodes
    fprintf('Tree #%g Dijkstra SHORTEST_PATH (%g, %g) : ',STindex,node,STsource);
    fprintf('%g', node);
    while(node ~= STsource)
      node = PredecessorNode(1,node);
      fprintf(',%g', node);
```

```
    end
    fprintf('\n');
  end
  % Initializing a New Graph to Plot the selected Steiner Tree
  figure('Name','EMECA Final Steiner Tree');
  PlotGraph(Nodes,Location,Topology);
  STVisitedCoordinates=[0 0 0 0];
  % Plotting Steiner Tree
  for index = 2:STcolumn
    Flag=0; % Flag to indicate if found edge visited before
    node = SteinerTree(STindex,index);
    fprintf('Tree #%g Steiner SHORTEST_PATH (%g, %g) : ',STindex,node,STsource);
    fprintf('%g', node);
    while(node ~= STsource)
      PlotEdge(Location,PredecessorNode,node);
      node1 = PredecessorNode(1,node);
      x1 = Location(node1,1);
      y1 = Location(node1,2);
      x2 = Location(node,1);
      y2 = Location(node,2);
      [r,c]=size(STVisitedCoordinates);
      for STCIndex=1:r
        if
((x1==STVisitedCoordinates(STCIndex,1))&&(y1==STVisitedCoordinates(STCIndex,2))&&(x2==ST
VisitedCoordinates(STCIndex,3))&&(y2==STVisitedCoordinates(STCIndex,4)))
          Flag=1;
        end
      end
      if Flag==0
```

```
            STVisitedCoordinates = [STVisitedCoordinates ; x1 y1 x2 y2 ];
        end
        node = PredecessorNode(1,node);
        fprintf(',%g', node);
        Flag=0;
    end
    fprintf('\n');
  end
  title 'EMECA Final Steiner Tree';
end
```

EMECA Results

EMECA Graphs

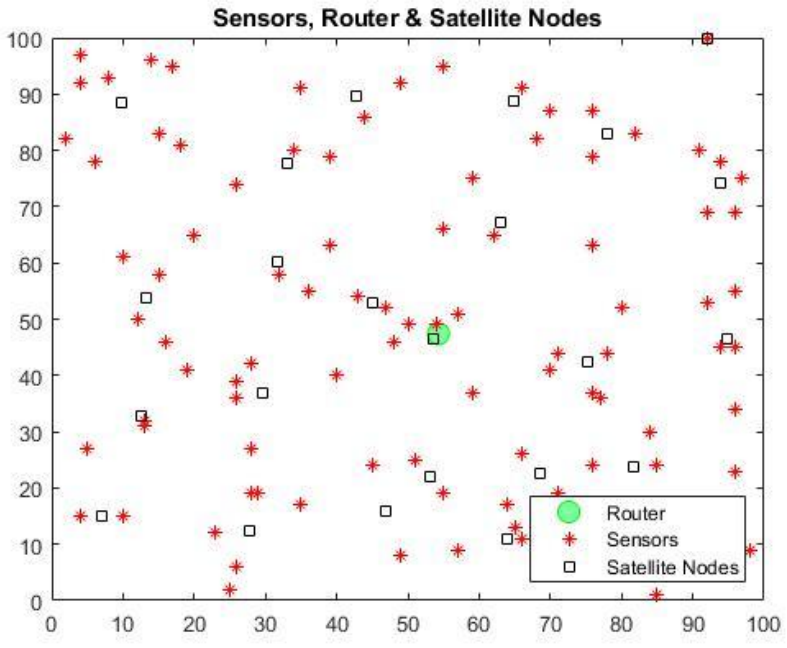

Figure 39 - EMECA Algorithm – Sensors, Routers and Satellite Nodes

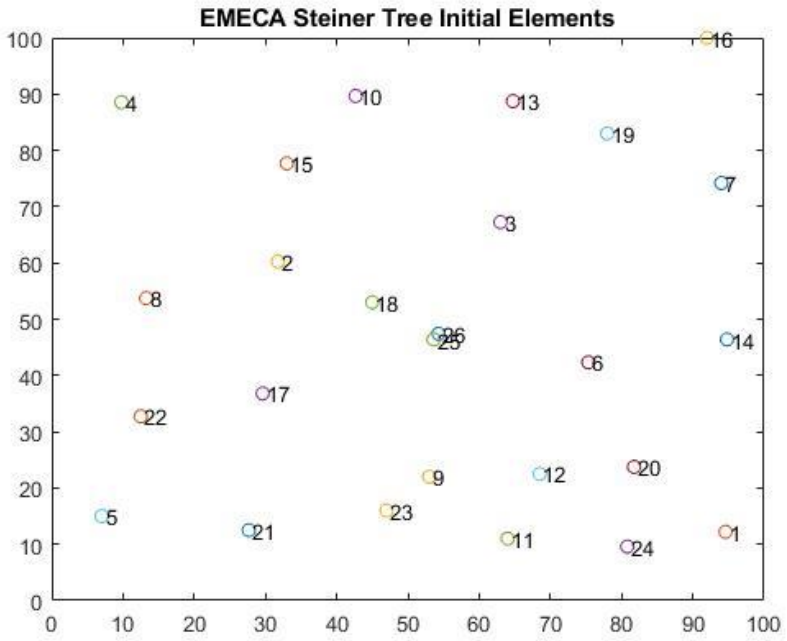

Figure 40 - EMECA Algorithm – Steiner Tree Initial Elements

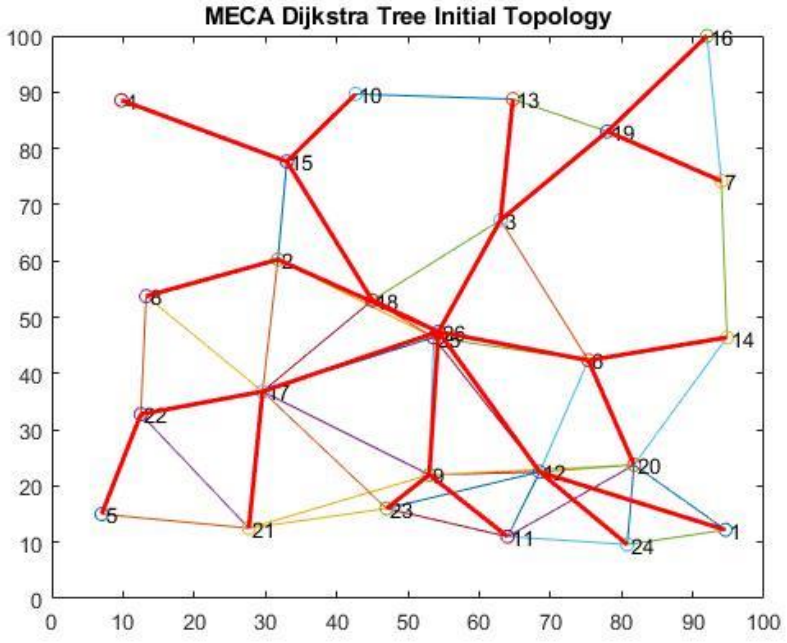

Figure 41 - EMECA Algorithm – Dijkstra Tree Inital Topology

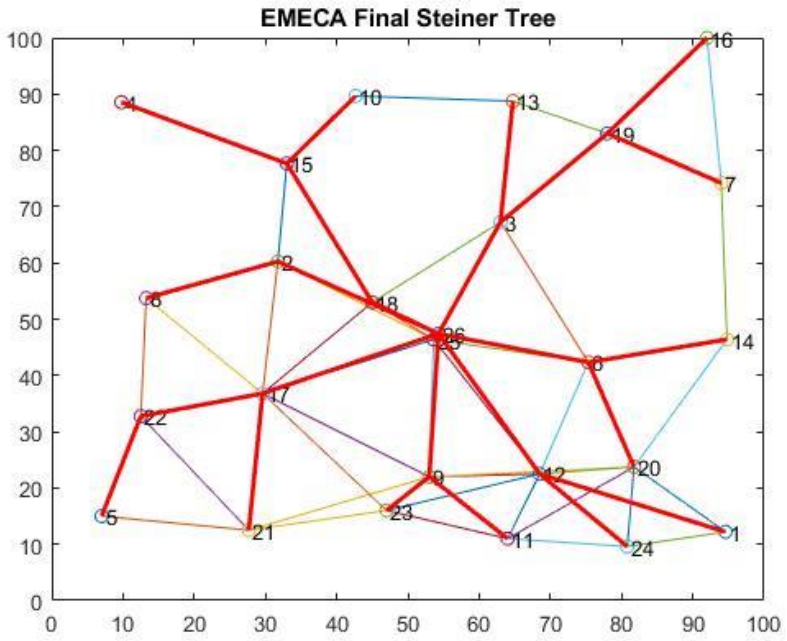

Figure 42 - EMECA Algorithm – Final Steiner Tree

EMECA MATLAB Command Window Output

Tree #1 Dijkstra SHORTEST_PATH (1, 26) : 1,12,26

Tree #1 Dijkstra SHORTEST_PATH (2, 26) : 2,26

Tree #1 Dijkstra SHORTEST_PATH (3, 26) : 3,26

Tree #1 Dijkstra SHORTEST_PATH (4, 26) : 4,15,18,26

Tree #1 Dijkstra SHORTEST_PATH (5, 26) : 5,22,17,26

Tree #1 Dijkstra SHORTEST_PATH (6, 26) : 6,26

Tree #1 Dijkstra SHORTEST_PATH (7, 26) : 7,19,3,26

Tree #1 Dijkstra SHORTEST_PATH (8, 26) : 8,2,26

Tree #1 Dijkstra SHORTEST_PATH (9, 26) : 9,26

Tree #1 Dijkstra SHORTEST_PATH (10, 26) : 10,15,18,26

Tree #1 Dijkstra SHORTEST_PATH (11, 26) : 11,9,26

Tree #1 Dijkstra SHORTEST_PATH (12, 26) : 12,26

Tree #1 Dijkstra SHORTEST_PATH (13, 26) : 13,3,26

Tree #1 Dijkstra SHORTEST_PATH (14, 26) : 14,6,26

Tree #1 Dijkstra SHORTEST_PATH (15, 26) : 15,18,26

Tree #1 Dijkstra SHORTEST_PATH (16, 26) : 16,19,3,26

Tree #1 Dijkstra SHORTEST_PATH (17, 26) : 17,26

Tree #1 Dijkstra SHORTEST_PATH (18, 26) : 18,26

Tree #1 Dijkstra SHORTEST_PATH (19, 26) : 19,3,26

Tree #1 Dijkstra SHORTEST_PATH (20, 26) : 20,6,26

Tree #1 Dijkstra SHORTEST_PATH (21, 26) : 21,17,26

Tree #1 Dijkstra SHORTEST_PATH (22, 26) : 22,17,26

Tree #1 Dijkstra SHORTEST_PATH (23, 26) : 23,9,26

Tree #1 Dijkstra SHORTEST_PATH (24, 26) : 24,12,26

Tree #1 Dijkstra SHORTEST_PATH (25, 26) : 25,26

Tree #1 Dijkstra SHORTEST_PATH (26, 26) : 26

Tree #1 Steiner SHORTEST_PATH (1, 26) : 1,12,26

Tree #1 Steiner SHORTEST_PATH (2, 26) : 2,26

Tree #1 Steiner SHORTEST_PATH (3, 26) : 3,26

Tree #1 Steiner SHORTEST_PATH (4, 26) : 4,15,18,26

Tree #1 Steiner SHORTEST_PATH (5, 26) : 5,22,17,26

Tree #1 Steiner SHORTEST_PATH (6, 26) : 6,26

Tree #1 Steiner SHORTEST_PATH (7, 26) : 7,19,3,26

Tree #1 Steiner SHORTEST_PATH (8, 26) : 8,2,26

Tree #1 Steiner SHORTEST_PATH (9, 26) : 9,26

Tree #1 Steiner SHORTEST_PATH (10, 26) : 10,15,18,26

Tree #1 Steiner SHORTEST_PATH (11, 26) : 11,9,26

Tree #1 Steiner SHORTEST_PATH (12, 26) : 12,26

Tree #1 Steiner SHORTEST_PATH (13, 26) : 13,3,26

Tree #1 Steiner SHORTEST_PATH (14, 26) : 14,6,26

Tree #1 Steiner SHORTEST_PATH (15, 26) : 15,18,26

Tree #1 Steiner SHORTEST_PATH (16, 26) : 16,19,3,26

Tree #1 Steiner SHORTEST_PATH (17, 26) : 17,26

Tree #1 Steiner SHORTEST_PATH (18, 26) : 18,26

Tree #1 Steiner SHORTEST_PATH (19, 26) : 19,3,26

Tree #1 Steiner SHORTEST_PATH (20, 26) : 20,6,26

Tree #1 Steiner SHORTEST_PATH (21, 26) : 21,17,26

Tree #1 Steiner SHORTEST_PATH (22, 26) : 22,17,26

Tree #1 Steiner SHORTEST_PATH (23, 26) : 23,9,26

Tree #1 Steiner SHORTEST_PATH (24, 26) : 24,12,26

Tree #1 Steiner SHORTEST_PATH (25, 26) : 25,26

>>

EMECA vs MECA

As per above code and output, we have verified that EMECA using the double clustering at the beginning will only need to run Dijkstra Shortest Path Algorithm to get the optimal Mesh-Based Wi-Fi System's Satellite Nodes location. This means that running Steiner Shortest Path Algorithm could only be used to verify the output, but not as a means to reach the optimal location for Mesh-Based Wi-Fi System's Router and Satellite Nodes. All of this is made possible by the rapid exponentially changing and evolving wireless technologies innovations and advancements in communications industry allowing us to revisit and enhance older models to match our current future needs of faster execution times and enhanced data processing models.

CHAPTER 6 CONCLUSION

In the above Industry 4.0 Information and Communication Technology Green Digital Foundation from Sensors to Services thesis contribution, we have managed to provide an unprecedented design of how the Fourth Industrial Revolution technologies, i.e., Cyber-Physical Production Systems (CPPS), Industrial Internet of Things (IIoT), Hybrid Cloud Manufacturing (HCMfg) and Manufacturing as a Service (MfgaaS), use the Third Platform foundation, built on Cloud, Big Data, Mobile, and Social Applications and Services, to interconnect, communicate, exchange messages/data, and store, retrieve and process information. In addition, we have modified and enhanced an IEEE Minimal Energy Consumption Algorithm (MECA) using the new technologies and resources made available in the past couple of years to deliver an Enhanced MECA.

In the first part of the contribution our extensible Service-Oriented Architecture Design in Chapter 4, we highlighted the main component Servers to be deployed on the Hybrid Cloud, made of interconnected Private “Local” and Public Clouds, in addition to each Server application and service modules, where we have stressed on CPPS, HCMfg, and MfgaaS. All designed system modules are loosely coupled allowing the DevOps engineers to dynamically at runtime with minimal to zero downtime install new modules, update existing ones, migrate between Servers, and, overall, easily and efficiently administrate the SOA designed system. We showed how the Cyber-Physical Connector Servers “Device Servers” connect the Cyber-Physical Systems (CPS) to the Hybrid Manufacturing Cloud Backend Database and Processing Servers creating a high-level feedback loop, made of the processed data stored as meaningful information on the database combined with the Backend Artificial Intelligence Analytics and

Frontend Human-Interactive Analytics Server extensible modules, to deliver quality manufacturing and highly optimized production performance.

In second contribution part Chapter 5 focused on Industry 4.0 IIoT theoretical and practical hands-on green algorithm reimplementation, modification/enhancement, development, and testing, we introduced the IEEE MECA prepared to solve an NP-Hard problem using a Green IoT Optimization Model focused on minimizing three main algorithmic constraints, i.e., energy consumption, data flow between devices, and overall system budget. The MECA uses K-means Clustering, and Dijkstra and Steiner Trees to simulate the optimization model in MATLAB and find the best possible resolution to the problem. We successfully reconstructed, coded and tested the MECA optimization model; then, we found an enhancement model based on newly introduced technological advancement models, where we replace the serial-connected relays losing an average of half of the bandwidth the further they are from the Base Station with a Mesh-Based WiFi System consisting of a Router replacing Base Station and Satellite Nodes replacing the Relays. Additionally, we determine the location of the Router using K-means clustering twice, where it finds in the first run the best location for Satellite Nodes followed by a second run for optimal location of Router. Moreover, we minimize the node to node communication through the SOA design introduced in Chapter 4 allowing for the use of less complex and cheaper mounted chipsets minimizing both data flow and budget constraints. Finally, using Mesh-Based WiFi we have only one network Service Set Identifier (SSID) “network name” allowing Radio-Frequency Identification (RFID) industrial parts and IoT devices moving from one node to the other not to register into too many SSIDs as per previously MECA design reducing administration time translated into overall budget reduction.

References

Abele, Eberhard, et al. “Learning Factories for Future Oriented Research and Education in Manufacturing.” *CIRP Annals - Manufacturing Technology*, vol. 66, no. 2, CIRP, 2017, pp. 803–26, doi:10.1016/j.cirp.2017.05.005.

Al-Anbagi, Irfan, et al. “A Reliable IEEE 802.15.4 Model for Cyber Physical Power Grid Monitoring Systems.” *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 2, Dec. 2013, pp. 258–72, doi:10.1109/TETC.2013.2281192.

Antonini, Alessio, et al. “Security Challenges in Building Automation and SCADA.” *Proceedings - International Carnahan Conference on Security Technology*, vol. 2014–Octob, no. October, IEEE, 2014, pp. 1–6, doi:10.1109/CCST.2014.6986996.

Azimi, Iman, et al. “HiCH.” *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 5s, ACM, Sept. 2017, pp. 1–20, doi:10.1145/3126501.

Bender, Beate, et al. “Learning Factory 2.0 -Integrated View of Product Development and Production.” *Procedia CIRP*, vol. 32, no. Clf, Elsevier B.V., 2015, pp. 98–103, doi:10.1016/j.procir.2015.02.226.

Brant, Anne, and Murali M. Sundaram. “A Novel System for Cloud-Based Micro Additive Manufacturing of Metal Structures.” *Journal of Manufacturing Processes*, vol. 20, Elsevier, Oct. 2015, pp. 478–84, doi:10.1016/j.jmapro.2015.06.020.

Broy, Manfred, et al. “Cyber-Physical Systems: Imminent Challenges.” *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence*

and Lecture Notes in Bioinformatics), vol. 7539 LNCS, Springer, Berlin, Heidelberg, 2012, pp. 1–28, doi:10.1007/978-3-642-34059-8_1.

Bu, Lei, et al. "Demo Abstract: BACH OL - Modeling and Verification of Cyber-Physical Systems Online." *Proceedings - 2012 IEEE/ACM 3rd International Conference on Cyber-Physical Systems, ICCPS 2012*, IEEE, 2012, p. 222, doi:10.1109/ICCPS.2012.43.

Cachay, Jan, et al. "Study on Action-Oriented Learning with a Learning Factory Approach." *Procedia - Social and Behavioral Sciences*, vol. 55, 2012, pp. 1144–53, doi:10.1016/j.sbspro.2012.09.608.

Cárenas, Alvaro A., and Amin. "Challenges for Securing Cyber Physical Systems." *Prevention*, 2009, p. submitted} @techreport{LEE:2007:ID2217, author=, doi:10.1.1.152.5198.

Challenge, Leadership Under. "Information Technology R&D in a Competitive World." *President's Council of Advisors on Science and Technology*, 2007, pp. 31–43.

Chen, Lixing, et al. "Adaptive Fog Configuration for the Industrial Internet of Things." *IEEE Transactions on Industrial Informatics*, 2018, pp. 1–1, doi:10.1109/TII.2018.2846549.

Cheng, St, and Ty Chang. "A Cyber Physical System Model Using Genetic Algorithm for Actuators Control." *Consumer Electronics, Communications …*, IEEE, Apr. 2012, pp. 2269–72, doi:10.1109/CECNet.2012.6201857.

Chryssolouris, George, et al. "Manufacturing Systems: Skills & Competencies for the

Future." *Procedia CIRP*, vol. 7, 2013, pp. 17–24, doi:10.1016/j.procir.2013.05.004.

DeMarco, C. L., et al. "The Potential for Malicious Control in a Competitive Power Systems Environment." *{IEEE} Int.\ Conf.\ on Control Applications*, IEEE, 1996, pp. 462–67, doi:10.1109/CCA.1996.558870.

Deshmukh, Siddharth, et al. "State Estimation in Spatially Distributed Cyber-Physical Systems: Bounds on Critical Measurement Drop Rates." *Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCoSS 2013*, IEEE, 2013, pp. 157–64, doi:10.1109/DCOSS.2013.23.

Erol, Selim, et al. "Tangible Industry 4.0: A Scenario-Based Approach to Learning for the Future of Production." *Procedia CIRP*, vol. 54, Elsevier B.V., 2016, pp. 13–18, doi:10.1016/j.procir.2016.03.162.

Gao, Sheng, et al. "A Cross-Domain Recommendation Model for Cyber-Physical Systems." *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 2, Dec. 2013, pp. 384–93, doi:10.1109/TETC.2013.2274044.

Görke, Matthias, et al. "Employee Qualification by Digital Learning Games." *Procedia Manufacturing*, vol. 9, The Author(s), 2017, pp. 229–37, doi:10.1016/j.promfg.2017.04.040.

Gräßler, Iris, Alexander Pöhler, et al. "Creation of a Learning Factory for Cyber Physical Production Systems." *Procedia CIRP*, vol. 54, The Author(s), 2016, pp. 107–12, doi:10.1016/j.procir.2016.05.063.

Gräßler, Iris, Patrick Taplick, et al. "Educational Learning Factory of a Holistic Product

Creation Process." *Procedia CIRP*, vol. 54, The Author(s), 2016, pp. 141–46, doi:10.1016/j.procir.2016.05.103.

Hahn, Adam, et al. "Cyber-Physical Security Testbeds: Architecture, Application, and Evaluation for Smart Grid." *IEEE Transactions on Smart Grid*, vol. 4, no. 2, June 2013, pp. 847–55, doi:10.1109/TSG.2012.2226919.

Hu, Fei, et al. "Robust Cyber-Physical Systems: Concept, Models, and Implementation." *Future Generation Computer Systems*, vol. 56, North-Holland, Mar. 2016, pp. 449–75, doi:10.1016/j.future.2015.06.006.

Huang, Jun, et al. "A Novel Deployment Scheme for Green Internet of Things." *IEEE Internet of Things Journal*, vol. 1, no. 2, 2014, pp. 196–205, doi:10.1109/JIOT.2014.2301819.

Jorgensen, Jens E., et al. "The Learning Factory - Curriculum Integration of Design and Manfacturing." *Proceedings of the Fourth World Conference on Engineering Education*, no. October, 1995, pp. 1–7, https://pdfs.semanticscholar.org/feec/6aee8134ef124e5db19e5b10041dda41c3fd.pdf.

Jovane, Francesco, et al. *The ManuFuture Road: Towards Competitive and Sustainable High-Adding-Value Manufacturing*. Springer Science & Business Media, 2008.

Juraschek, Max, et al. "Experiencing Closed Loop Manufacturing in a Learning Environment." *Procedia Manufacturing*, vol. 9, The Author(s), 2017, pp. 57–64, doi:10.1016/j.promfg.2017.04.046.

Karre, Hugo, et al. "Transition towards an Industry 4.0 State of the LeanLab at Graz

University of Technology." *Procedia Manufacturing*, vol. 9, The Author(s), 2017, pp. 206–13, doi:10.1016/j.promfg.2017.04.006.

Kondo, Derrick, et al. "Cost-Benefit Analysis of Cloud Computing versus Desktop Grids." *2009 IEEE International Symposium on Parallel & Distributed Processing*, IEEE, 2009, pp. 1–12, doi:10.1109/IPDPS.2009.5160911.

Lamancusa, John S., et al. "2006 Bernard M. Gordon Prize Lecture*: The Learning Factory: Industry-Partnered Active Learning." *Journal of Engineering Education*, vol. 97, no. 1, 2008, pp. 5–11, doi:10.1002/j.2168-9830.2008.tb00949.x.

Lee, Insup, et al. "Challenges and Research Directions in Medical Cyber-Physical Systems." *Proceedings of the IEEE*, vol. 100, no. 1, Jan. 2012, pp. 75–90, doi:10.1109/JPROC.2011.2165270.

Lee, Jay, Behrad Bagheri, et al. "A Cyber-Physical Systems Architecture for Industry 4.0-Based Manufacturing Systems." *Manufacturing Letters*, vol. 3, Elsevier, Jan. 2015, pp. 18–23, doi:10.1016/j.mfglet.2014.12.001.

Lee, Jay, Edzel Lapira, et al. "Recent Advances and Trends in Predictive Manufacturing Systems in Big Data Environment." *Manufacturing Letters*, vol. 1, no. 1, Elsevier, Oct. 2013, pp. 38–41, doi:10.1016/j.mfglet.2013.09.005.

Lenk, Alexander, et al. "What's inside the Cloud? An Architectural Map of the Cloud Landscape." *2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, IEEE, 2009, pp. 23–31, doi:10.1109/CLOUD.2009.5071529.

Li, Ming, and Pan Li. "Crowdsourcing in Cyber-Physical Systems: Stochastic Optimization with Strong Stability." *IEEE Transactions on Emerging Topics in*

Computing, vol. 1, no. 2, Dec. 2013, pp. 218–31, doi:10.1109/TETC.2013.2273358.

Lin, Chun Cheng, and Jhih Wun Yang. "Cost-Efficient Deployment of Fog Computing Systems at Logistics Centers in Industry 4.0." *IEEE Transactions on Industrial Informatics*, vol. 3203, no. c, 2018, doi:10.1109/TII.2018.2827920.

Ma, Zhiqiang, et al. "Object-Oriented Petri Nets Based Formal Modeling for High-Confidence Cyber-Physical Systems." *2012 International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2012*, IEEE, 2012, pp. 1–4, doi:10.1109/WiCOM.2012.6478590.

Mitchell, Robert, and Ing-Ray Chen. "Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems." *IEEE Transactions on Reliability*, vol. 62, no. 1, Mar. 2013, pp. 199–210, doi:10.1109/TR.2013.2240891.

Mo, Yilin, et al. "Cyber-Physical Security of a Smart Grid Infrastructure." *Proceedings of the IEEE*, vol. 100, no. 1, Jan. 2012, pp. 195–209, doi:10.1109/JPROC.2011.2161428.

Monostori, L., et al. "Cyber-Physical Systems in Manufacturing." *CIRP Annals - Manufacturing Technology*, vol. 65, no. 2, Elsevier, Jan. 2016, pp. 621–41, doi:10.1016/j.cirp.2016.06.005.

Moreno, Javier, et al. "Unified and Comprehensive Electronic System Level, Network and Physics Simulation for Wirelessly Networked Cyber Physical Systems." *Forum on Specification and Design Languages (FDL), 2012*, 2012, pp. 68–74.

Peng, Chengzhang, and Zejun Jiang. "Building a Cloud Storage Service System."

Procedia Environmental Sciences, vol. 10, Elsevier, Jan. 2011, pp. 691–96, doi:10.1016/J.PROENV.2011.09.111.

Prinz, Christopher, et al. "Learning Factory Modules for Smart Factories in Industrie 4.0." *Procedia CIRP*, vol. 54, The Author(s), 2016, pp. 113–18, doi:10.1016/j.procir.2016.05.105.

Quaglia, Davide. "Communications in Cyber-Physical Systems." *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*, IEEE, 2013, pp. 1–1, doi:10.1109/MECO.2013.6601382.

---. "Cyber-Physical Systems: Modeling, Simulation, Design and Validation." *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*, IEEE, June 2013, pp. 1–2, doi:10.1109/MECO.2013.6601389.

Rajamaki, Jyri, et al. "Resilience of Cyber-Physical System: A Case Study of Safe School Environment." *2012 European Intelligence and Security Informatics Conference*, IEEE, 2012, pp. 285–285, doi:10.1109/EISIC.2012.10.

Rajhans, Akshay, et al. "An Architectural Approach to the Design and Analysis of Cyber-Physical Systems." *Electronic Communications of the EASST*, vol. 21, no. 0, Nov. 2009, doi:10.14279/tuj.eceasst.21.286.278.

Rajkumar, Ragunathan (Raj), et al. "Cyber-Physical Systems." *Proceedings of the 47th Design Automation Conference on - DAC '10*, ACM Press, 2010, p. 731, doi:10.1145/1837274.1837461.

Ravindran, Kaliappa, and Mohammad Rabby. "Cyber-Physical Systems Based Modeling of Adaptation Intelligence in Network Systems." *2011 IEEE*

International Conference on Systems, Man, and Cybernetics, IEEE, Oct. 2011, pp. 2737–42, doi:10.1109/ICSMC.2011.6084087.

Reith, S. "Außerbetriebliche CIM-Schulung in Der „Lernfabrik"." *Produktionsforum'88. Die CIM-Fähige Fabrik. Springer, Berlin, Heidelberg*, 1988, pp. 581–601.

Rentzos, L., et al. "Integrating Manufacturing Education with Industrial Practice Using Teaching Factory Paradigm: A Construction Equipment Application." *Procedia CIRP*, vol. 17, 2014, pp. 189–94, doi:10.1016/j.procir.2014.01.126.

Samad, Tariq. "Control Systems and the Internet of Things [Technical Activities]." *IEEE Control Systems*, vol. 36, no. 1, Feb. 2016, pp. 13–16, doi:10.1109/MCS.2015.2495022.

Schuh, Günther, Jan Philipp Prote, et al. "Classification of a Hybrid Production Infrastructure in a Learning Factory Morphology." *Procedia Manufacturing*, vol. 9, The Author(s), 2017, pp. 17–24, doi:10.1016/j.promfg.2017.04.007.

Schuh, Günther, Thomas Gartzen, et al. "Promoting Work-Based Learning through Industry 4.0." *Procedia CIRP*, vol. 32, no. Clf, 2015, pp. 82–87, doi:10.1016/j.procir.2015.02.213.

Seitz, Kai Frederic, and Peter Nyhuis. "Cyber-Physical Production Systems Combined with Logistic Models-a Learning Factory Concept for an Improved Production Planning and Control." *Procedia CIRP*, vol. 32, no. Clf, Elsevier B.V., 2015, pp. 92–97, doi:10.1016/j.procir.2015.02.220.

Simons, Stephan, et al. "Learning in the AutFab – The Fully Automated Industrie 4.0

Learning Factory of the University of Applied Sciences Darmstadt." *Procedia Manufacturing*, vol. 9, The Author(s), 2017, pp. 81–88, doi:10.1016/j.promfg.2017.04.023.

Suto, Katsuya, et al. "An Energy-Efficient and Delay-Aware Wireless Computing System for Industrial Wireless Sensor Networks." *IEEE Access*, vol. 3, 2015, pp. 1026–35, doi:10.1109/ACCESS.2015.2443171.

Thiede, Sebastian, et al. "Implementing Cyber-Physical Production Systems in Learning Factories." *Procedia CIRP*, vol. 54, Elsevier B.V., 2016, pp. 7–12, doi:10.1016/j.procir.2016.04.098.

Truong, Hong Linh, and Schahram Dustdar. "Composable Cost Estimation and Monitoring for Computational Applications in Cloud Computing Environments." *Procedia Computer Science*, vol. 1, no. 1, Elsevier, 2010, pp. 2175–84, doi:10.1016/j.procs.2010.04.243.

Vermesan, O., Friess, P., Guillemin, P., Gusmeroli, S., Sundmaeker, H., Bassi, A., Jubert, I.S., Mazura, M., Harrison, M., Eisenhauer, M. and Doody, P. *Internet of Things Strategic Research Roadmap.* Internet of Things-Global Technological and Societal Trends, 2011.

Wan, Jiafu, et al. "From Machine-to-Machine Communications towards Cyber-Physical Systems." *Computer Science and Information Systems*, vol. 10, no. 3, 2013, pp. 1105–28, doi:10.2298/CSIS120326018W.

Wank, Andreas, et al. "Using a Learning Factory Approach to Transfer Industrie 4.0 Approaches to Small- and Medium-Sized Enterprises." *Procedia CIRP*, vol. 54,

The Author(s), 2016, pp. 89–94, doi:10.1016/j.procir.2016.05.068.

Xu, Jie, et al. “Online Learning for Offloading and Autoscaling in Energy Harvesting Mobile Edge Computing.Pdf.” *Ieee Transactions on Cognitive Communications and Networking*, vol. 3, no. 3, Sept. 2017, pp. 361–73, doi:10.1109/TCCN.2017.2725277.

Yang, Lei, et al. “Cost Aware Service Placement and Load Dispatching in Mobile Cloud Systems.” *IEEE Transactions on Computers*, vol. 65, no. 5, May 2016, pp. 1440–52, doi:10.1109/TC.2015.2435781.

Zanero, Stefano. “Cyber-Physical Systems.” *Computer*, vol. 50, no. 4, 2017, pp. 14–16, doi:10.1109/MC.2017.105.

Zhai, Xiaoxiang, et al. “A Unified Modeling and Verifying Framework for Cyber Physical Systems.” *Proceedings - International Conference on Quality Software*, IEEE, 2012, pp. 128–31, doi:10.1109/QSIC.2012.11.