FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Software System Engineering for a Cyber Physical Production Systems

Arjun Sah

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Mestrado em Engenharia de Software

Supervisor: Gil Gonçalves

September 21, 2020

Software System Engineering for a Cyber Physical Production Systems

Arjun Sah

Mestrado em Engenharia de Software

September 21, 2020

Abstract

Cyber physical production systems are the future of production systems not only in Europe but in the entire world. It brings with itself huge benefits and popularly attributes to Industry 4.0 also. These are automated systems where physical systems are monitored and controlled by computer-based algorithms in real time. Traditional systems have certain disadvantages and are limited in terms of hours of operation as it is governed by manpower and the type of products that can be produced without making much changes in the production configuration and the speed of production of products. In Europe, a lot of research is going on, particularly in Germany and in the United states too for upgrading major physical systems and manufacturing systems. Some examples of such systems are smart factory, smart grid, autonomous automobile systems, automatic pilot avionics, robotics systems etc.

The main goal of this thesis is to define a set of methodologies for easing the process of implementation of the CPPS(cyber physical production systems) system on small and medium industries so that the adoption rate for such industries can be high. There is no methodology yet particularly for CPPS systems for small and medium industries, although we have methodologies in place for large industries.

In order to do so, first study was done for challenges in developing a requirement engineering process in section 3 and how it is different from a typical software system. An approach has been developed based on existing information available on large systems and CPPS and some software engineering frameworks like MODAF and TOGAF. A proposal for the process and some diagrams and tools has been made in section 4.

To validate the proposed approach, we have taken a synthetic test case of a pizza production system and implemented all the approaches to transform it into a cyber physical production system right from requirement and UML diagrams to the final function block approach.

With this set of approaches, there is now a basis for software development methodology for small and medium industries particularly. With these approaches the adoption rate can be really high for such industries bringing out traditional industries more to the 21st century forefront.

Keywords: software engineering, requirement engineering, cyber physical production systems

Resumo

Os sistemas de produção ciber-físicos são o futuro dos sistemas de produção não apenas na Europa, mas em todo o mundo. Esta visão traz consigo enormes benefícios e é também associada à Indústria 4.0. Estes são sistemas automatizados onde os sistemas físicos são monitorizados e controlados por algoritmos baseados em computador em tempo real. Os sistemas tradicionais têm certas desvantagens e são limitados em termos de horas de operação, pois são regidos pela mão de obra e pelo tipo de produtos que podem ser produzidos sem fazer muitas alterações na configuração de produção e na velocidade de produção dos produtos. Na Europa, há um volume grande de investigação em andamento, especialmente na Alemanha e também nos Estados Unidos para atualizar os principais sistemas físicos e sistemas de manufatura. Alguns exemplos de tais sistemas são fábrica inteligente, rede inteligente, sistemas autónomos, sistemas robóticos, etc. O objetivo principal desta tese é definir um conjunto de metodologias para facilitar o processo de implantação do sistema CPPS (cyber Physical Production Systems) em pequenas e médias indústrias de forma que a taxa de adoção por tais indústrias possa ser elevada. Ainda não existe uma metodologia específica para sistemas CPPS para pequenas e médias indústrias, embora tenhamos metodologias em vigor para grandes indústrias.

Para fazer isso, o primeiro estudo foi feito para os desafios no desenvolvimento de um processo de engenharia de requisitos na seção 3 e verifica-se como ele é diferente de um sistema de software típico. Uma abordagem foi desenvolvida com base nas informações existentes disponíveis em grandes sistemas e CPPS e alguns frameworks de engenharia de software como MODAF e TOGAF. Uma proposta para o processo e alguns diagramas e ferramentas foi feita na seção 4.

Para validar a abordagem proposta, pegamos um caso de teste sintético de um sistema de produção de automáticos e implementamos todas as abordagens para transformá-lo em um sistema de produção ciber-físico desde os diagramas de requisitos e UML até a abordagem do bloco de funções final. Com esta abordagem, agora existe uma base para a metodologia de desenvolvimento de software para sistemas ciber-físicos pequenas e médias indústrias em particular. Com essas abordagens, a taxa de adoção pode ser muito alta para esses setores, trazendo os setores tradicionais mais para a vanguarda do século 21.

Keywords: engenharia de software, engenharia de requisitos, sistemas cibernéticos de produção física

Acknowledgements

I want to thank Professor Ana Paiva, Professor Nuno Flores and Professor Gil Gonçalves for being the amazing professors that they have been all through my Masters.

Right from the time I joined and started learning from the gem of the teachers that they have been, was very educative.

Every semester that i have gone through I learned something new that was missing in me earlier and after learning all those things I feel very proud. Professor Ana Paiva was like the most influential professor for me as she has always focused on learning and doing the things no matter what.

Professor Nuno Flores was the best teacher to sit and hear in the classroom. He will make the teaching full of fun and knowledge. He is always fun to be in the classroom without an ounce of sleep in the eyes.

To my friends, a big thank you, Joao santos, Erica Lima and Mila Kisialova for being the best buddies a person could ask for. All those good and bad moments, the way Erica used to stand with me was very kind and being the best person that she is. Joao Santos, a person of intelligence and high caliber like a sharp person focused on work and study together, you have taught me a lot in life that if you are focused then you can do anything.

Professor Gil, I first got to know during Requirement engineering classes and in the end, I did my thesis under him. The aura and calmness he carries is unlike I have seen. The calmness with which he guided me during my thesis in all the hard phases and broke down difficult things to chunks. I am really grateful to him for sharing all the knowledge and support with me.

"Faithless is he that says farewell when the road darkens."

J.R.R Tolkien

Contents

List of Figures

List of Tables

Abbreviations

CPS	Cyber Physical System
CPPS	Cyber Physical Production System
MODAF	Ministry of Defence Architectural Framework
TOGAF	The Open Group Architecture Framework
UML	Unified Modelling Language
NSF	National Science Foundation
NIT	Networking and Information technology
PCAST	President's Council of Advisors on Science and Technology
DFKI	The German Research Center for Artificial Intelligence
IOT	Internet Of Things
CPUs	Central Processing Units
PICs	Peripheral Interface
SoC	System on Chip
3D	three dimension
CAD	Computer Aided Design
PLM	Product Life cycle Management
SO-PC-PRO	Subject-Orientation for People-Centered Production
MIGOODS	Manufacturing Intelligence for Consumer Goods
ACMN	Automation Competency Model Network
SysML	System Modelling Language
OMG	Object Management Group
IEC	International Electrotechnical commission
FB	Function block
SISO	single input single output
MIMO	Multiple input Multiple output
BDD	Block Definition diagram
IBD	Internal block diagram
QoS	Quality of Service
OMG	Object Model Group
DAE	Differential Algebraic Equation
MDD	Model Driven Development
ADM	Architectural Development Methods
SEBok	Guide to the systems engineering Body of Knowledge
SE	Software Engineering
DIG2	The Digital and Intelligent Industry Lab
IIoT	Industrial Internet of things
AGVs	Automated Guided Vehicles
MES	Manufacturing Execution System
MTS	Mechatronic Systems
CCM	Corba component model

Chapter 1

Introduction

Cyber physical systems (CPS) is the future of all the automated systems where physical systems are monitored and controlled by computer-based algorithms in real time [5]. The software components and physical components are associated on a deeper level and exhibit dynamic characteristics and behaviors that change with contexts. Cyber Physical production systems (CPPS) are the CPS systems for production systems, used in factories and production units. Traditional systems have limits in terms of hours of operation as it is governed by manpower and the kind of products that can be produced without making much changes in the production configuration and the speed of production of products. The concept of cyber physical production systems is based on the perception that it is much efficient, can work continuously as long as it is required to function and value-added services result from technological evolution [17].

Cyber Physical systems are the latest buzz word in the 21st century which is very frequently interchanged by industry 4.0. In Europe, there is a lot of research going on, in Germany and in the United states too for technologically advancing all the major physical systems as well as manufacturing systems too. Some of the most popular examples are smart factory, smart grid, autonomous automobile systems, automatic pilot avionics, robotics systems etc.

Cyber physical systems involve multiple disciplines to come and work together in sync to have the results that were not possible earlier. It involves the tranciplinary approach of merging theories from cybernetics, mechatronics, design and process science [5]. A lot of research is going on and a lot of more research is needed to explore the full potential of this 21st century's new technology.

Some of the major advantages of CPS/CPPS can be thought of:

1. Optimized for efficient production processes;
2. product customization is highly optimized;
3. Efficiently using resources for maximum production;
4. Production process is centered around people as explained in [16]

1.1 Context

This is an era of the fourth industrial revolution where all the manufacturing and production systems are getting more and more advanced and automated and everything is getting connected and autonomous. This is popularly known as Industry 4.0. Since it is an integration of the physical world and cyber world and involves computation, networking and physical processes it can change the face of how industries have worked till now and can define a new way of working. Since industry is moving towards this direction, it can become a very strong foundation for our critical infrastructures and improve the services provided in many sectors. Many sectors can achieve huge advances such as health care, emergency response, traffic flow management etc. Nowadays we hear of smart cities, smart grids," smart" anything which can be cars, manufacturing, buildings, hospitals, homes etc. All these involve the usage of cyber physical systems. The United States of America first coined the term CPS at the National Science Foundation (NSF) around 2006 recognizing the need for a future networking and information technology (NIT) in the 2007 report of the President's Council of Advisors on Science and Technology (PCAST) and has been actively funding research towards exploiting its full potential [18].

The German Research Center for Artificial Intelligence (DFKI) in Bremen is working on cyber physical systems for the technical development of intelligent, networked systems. Prof. Dr. Rolf Drechsler is directing the department towards the research. The research is towards safety, security and correctness of these systems which are growing in complexity and are used in many crucial systems [6]. Scientists at the research center are collaborating with the Group of Computer Architecture at the university of Bremen. The center of DFKI, in Kaiserslautern, Saarbrucken and Bremen and a project office in Berlin, is the largest research center for artificial intelligence in the world [6].

1.2 Motivation and objectives

Since the cyber physical system is a very diverse and complex topic and is used for very large systems, there is a need to implement it for small systems or small industrial units and for that a requirement engineering process needs to be evolved. Due to the complexity of large systems a lot of variables are in motion and there is a need to model a lot of components as well as create a digital twin for the system so that hardware and software can work together. As well as the fluidity in behavior between hardware and software components leading to complexity in coordination and processes of components. All these complexities will be lesser for a small and medium systems as it involves less physical and mechanical components and thus less interaction among them and thus less modelling, although it does not mean the inherent complexity will be any lesser but relatively lesser in comparison. Implementing the approach for large and complex systems is harder and difficult due to the sheer scale and variables involved. Thus, there is a need to have an approach or strategy targeting specifically for small and medium industries so that industries can go for an easy way to adopt and take advantage of 21st century innovations. The requirement

engineering system for a static system already exists and is in place and the system also exists for a fully-fledged large scale cyber physical system. But for a small system it needs to be derived or proposed so that small industrial units can also take advantage of this new technology to reap its benefits and be more efficient in their operations. As the system is dynamic and many parts are constantly communicating to carry out a lot of dynamic tasks and hence requirements keep on changing and evolving as it involves a lot of engineering fields such as mechanical, electrical, electronics, embedded, computer science and data science too for displaying such a level of dynamic behaviors. The independent decision making of such systems with respect to when new products need to be developed to cater to market demands as well as replacing a system component that is going to be of no use in future based on past performance data is absolutely amazing. The requirement engineering process for such systems starts with SysML as the system needs to be modelled first on a hardware level as well as software level before such a dynamic system can be developed.

The main objective of this thesis is to develop an approach of software development methodology for cyber physical production systems of small and medium scale so that adoption rate can be very high.

1.3 Structure of the Thesis

This thesis is structured as follows. Chapter 2 describes state of the art, bibliographic research, cyber physical systems and difference between CPS and CPPS, CPPS research context, SysML, IEC61499. Chapter 3 describes challenges in requirement gathering for CPPS systems. Chapter 4 defines the methodology, proposal of the process, diagrams and tools. Chapter 5 presents the validation of the approach with a pizza production system with various SysML diagrams and UML diagrams.

Chapter 2

State of the art

In this chapter we present the existing research in the fields of Cyber physical systems, its design challenges and proposed solution. It also summarizes the system engineering principles and finally the requirement engineering challenges in cyber physical systems.

It gives an overview of the existing research that has been done in the field of cyber physical systems, cyber production systems and industry 4.0 and the problem that I am trying to solve by going through the existing research. The problem at hand is How to specify a cyber physical production system? I need to design a requirement engineering process for a cyber physical production system by building on existing systems engineering processes for complex systems. Cyber physical systems are the next generation thing which can integrate the physical world and cyber world completely through the usage of the internet/network. The concept of IOT (internet of things) comes handy here. We hear about smart cities in China, India, USA and in Europe and recently more about smart factories which can or have potential to work 24 hours a day, 7 days a week and 365 days a year uninterruptedly unlike humans which work only 8-10 hours a day in factories. And due to market demands various customer needs can be taken care of by producing different products from the same factory by Cyber physical production systems. It will boost the production and take care of market demands too and ultimately boost the economy of a region or country. This is sometimes known as the fourth industrial revolution which will not only change the way things are produced or done but completely change everything including us.

Some of the major applications identified and in use are smart grid, smart cities, smart factories, gas and oil pipeline monitoring and control [15]. These applications are so useful and productive that we cannot think about going back to manual handling. Nowadays software systems are hugely complicated itself and in CPPS (Cyber physical production system) there is a network of networks of different elements. IOT (Internet of things), embedded systems, cloud, sensors, storage are some of the elements for CPPS.

Manufacturing intelligence is what is being designed in CPPS as it can take decision of how much to produce and what varieties to produce. Unlike conventional production systems, CPPS

are system of systems where different disciplines are needed for their realization such as mechanical engineering, electrical engineering, and computer science as real world is represented by mechanical/electrical engineering and virtual world is represented by computer science. And there is a concept of digital twin of physical entities in the virtual world to smoothen the interaction and communication between different entities from real to virtual world and vice versa and since huge amount of data is exchanged between real and virtual world there is huge storage requirement and since communication needs to be very quick the hardware needs to be advanced as the current hardware or chips are not that much efficient. For communication the high-speed internet is required. The data that is coming from the physical world needs to be filtered and refined before allowing it to move further as it will contain noises that will impact decision making. The discipline of data science is very crucial as it will find useful information out of a huge amount of data and can help in proper decision making of the CPPS system. There is a huge opportunity in this CPPS field and because of the promise it holds there is a need to design an efficient requirement engineering process as the static requirement engineering process will not work here because of the dynamic nature of the system and changing requirements.

2.1 Bibliographic Research

The research mainly involved in understanding the cyber systems and their importance and usefulness in changing society and making our life better. Since times are changing and we are almost in fourth industrial revolution of industry that is popularly known as Industry 4.0 where everything is getting automated and connected to the internet and machines can take their own decisions in making our life better by producing in the most efficient way 24 *7 and 365 days a year.

2.1.1 Cyber-Physical Systems and Cyber-Physical Production Systems

Cyber physical systems (CPS) are those systems where cyber and physical systems are integrated and it is a general term. Cyber physical Production systems (CPPS) are those CPS systems which connect the production systems and are mostly associated with Industry 4.0. Cyber physical systems are considered to be upgraded versions of embedded systems [16]. As the embedded systems depend on chips and microcontrollers and since these are evolving and hence embedded systems too to take some advanced form. Earlier microcontrollers had limited speed of CPUs and data memories and limited communication ability. As the technologies advanced all these shortcomings were getting reduced and more built-in resources were integrated into microcontrollers to make it faster. For example, Microchip Peripheral Interface (PIC) family brought a lot of updated features in microcontrollers such as reduced instruction set in microcontrollers with high built in resources, more speed, Realtime clock that was built-in, wider data bus to handle 32-bit data and many more features [16]. These updated microcontrollers were used to build embedded systems which again were used to create many applications such as vehicle tracking, traffic control systems, home automation systems, in healthcare and in automating some processes in factories.

Inputs were gathered from sensors, processed by microcontrollers based on some embedded system algorithms and then given output for the desired results. Recent advancement gave the concept of a system on Chip (SoC) with more CPU power. The CPU power increased many folds from megahertz to gigahertz, internal storage and SD storage to 1000s of megabyte. These chips are equipped with wife, ethernet access points, Bluetooth and ZigBee's features which was not possible earlier [16]. These advanced features allowed it to connect to the internet/network and hence to cyber space and hence a nexus of cyber-physical systems was formed. CPS is the basis of next generation industry 4.0. The below chart represents the popular communication protocols that are in use and taken from [16] for reference.

2.1.2 Difference between CPS vs CPPS

CPPS is a case of CPS itself. Although the concept of CPPS may sound new. But the notion of CPPS with the same idea of solution existed globally in the past decades. As can be seen in some of the implementations such as Intelligent Manufacturing Systems, Biological Manufacturing Systems, Reconfigurable Manufacturing Systems, Digital Factory, Holonic Manufacturing Systems, Industrial Agents and in many more [15]. CPS is characterized by Computation, Control and Communication. While CPPS is characterized by Intelligence, connectedness and Responsiveness. The characteristics of CPS is extended in CPPS as can be shown in below diagram and taken from [15].

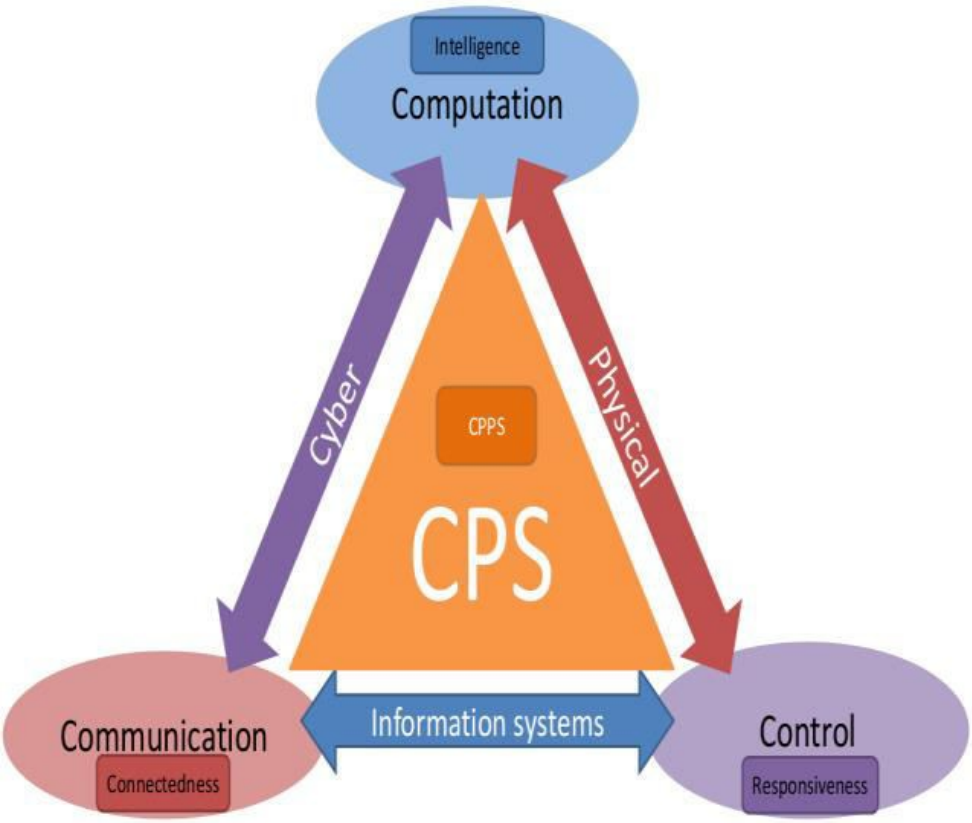

Figure 2.1: Basic capabilities of a CPS and the analogy to CPPS

As is displayed in the above diagram that computation is extended to Intelligence. Huge amounts of data collected in the form of big data is processed via data mining to look for some useful patterns in data and then applying analytics to find intelligence in data that can predict failure of some component in future and many more things. Communication in CPS is extended to connectedness in CPPS. Every component is connected with each other and communicates with each other to handle any unwanted response together. Control in CPS is extended to Responsiveness. Not only components are controlled but respond to any changes in internal/external parameters or signals changes.

2.1.3 CPPS research context

CPPS definition asks for the cooperation of CPS but many other research notions like knowledge management and decision making which form a huge part of research fields are missing. Research ideas like digital twins for dynamic simulation and forecasting are still not a part of overall research of CPPS. Other important fields that are missing includes learning from the existing data and adapting automatically. Since CPPS involves multiple technologies and ields the adaptation of systems to new technologies and major reconfiguration are not researched. Since CPPS is an extension of CPS and it brings different fields together for achieving a complex high-level system, still the

fundamental ideas that it consists of persisted long before CPPS itself. In the past decades several solutions achieving the same objective globally were being used already. Some of the examples are digital factory, Intelligent Manufacturing systems, Reconfigurable manufacturing systems etc. [15]. The major advantage of adopting CPPS over earlier similar solutions is that it can be accepted by a larger audience and under a single group/tag all the publications can be found making whole information more readable, clear and organized for the scientific community. A study of different international literature shows the main trends of research are concentrated around 3 main axes. The first axis points towards developing cutting edge technologies that can enhance production. The application of advanced technology in the manufacturing process can change the production scenario in the coming future and form the basis for next generation production systems. Current research focuses on smart materials or intelligent materials. Smart materials are those whose one or more property in a controlled manner can be changed significantly by external actions. These actions can be stress, moisture, electric or magnetic fields, light, temperature, pH, or chemical compound [13]. Most common applications can be found in sensors and actuators. Additive manufacturing is also one of the focus areas of research. It is most commonly referred to as 3D printing, where 3D object models are generated using CAD (computer aided design) [3]. It is mostly used to design lighter and stronger parts and systems. The second axis points to sustainable production. Sustainable production is the production of goods and services without harming nature, with less pollution, economically viable and rewarding to society as a whole. There were two ideas for sustaining production with reduced impact on the environment. The first was to increase the energy efficiency of the processes so that with less energy maximum work can be done. Some of the projects have implemented this idea like Development of Adaptive Production systems for Eco-efficient firing processes, Energy Efficient Manufacturing, Factory Eco-Friendly and energy efficient technologies and adaptive automation solutions [15]. The second idea is to develop closed loop PLM processes. It focusses on handling and keeping track of information of the whole product lifecycle with feedback from lifecycle phases wherever possible [21]. Implementing a PLM process takes a high level of close communication and coordination. By implementing the PLM process, we can identify and reuse useful parts of dysfunctional systems so as to reduce the burden on the environment of producing a new system altogether with all new components, thus reducing the environmental impact. The project that has used the PLM process is Sustainable and Efficient Production of Lightweight Solutions also known as Sup LIGHT project. The future production systems need the technology and human factors to work in cooperation. This CPPS process aims to be human centric production systems. Some of the projects that studied the above is Subject-Orientation for People-Centered Production also known as SO-PC-PRO. Other projects that implemented the efficient nexus of technology and human factor are Manufacturing through ergonomic and safe Anthropocentric adaptive workplaces for context aware factories in Europe known as MAN-MADE.

The final axis focusses on agility of production systems. Agility of such systems means adaptive over a period of time with respect to technology as well as market demands too. It also includes being intelligent and collaborative and interoperable. Being able to configure and reconfigure as

well as change and evolve is an important parameter. Projects such as GRACE, ARUM, PABADIS or ERRIC have shown significant advances in distributed or holonic technologies. Concepts such as distributed intelligence or intelligent products are common to distributed technologies. Mass customization was used in projects such as Manufacturing Intelligence for Consumer Goods (MI-GOODS). Good amount of work is being done on interoperability of systems as is displayed in projects such as ACMN and Linked Design. ACMN stands for Automation Competency Model Network and Linked Design stands for Linked Knowledge in Manufacturing, Engineering and Design for Next Generation Production.

2.2 SysML

SysML stands for system modelling language, and is a general-purpose language used by system designers to architect a system engineering application. Some of the features explained in [14] are:

- SysML supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems. These systems may include hardware, software, information, processes, personnel, and facilities.

- SysML is a dialect of UML 2, and is defined as a UML 2 Profile. (A UML Profile is a UML dialect that customizes the language via three mechanisms: Stereotypes, Tagged Values, and Constraints.)

- SysML is the foundation for model-based system engineering

The SysML was developed in 2003 by a project called SysML Partners' SysML Open Source Specification Project. It was in 2006 that it got adapted and adopted by OMG (Object Management Group) as OMG SysML.

For architecting any application, the first step is creating SysML diagrams to get the overview of how an application might look in its earlier forms before refining it to the form based on which we gather requirements and then development starts on the basis of that. SysML has many diagrams to design system level intricacies. The SysML is composed of nine diagram types. These are Requirement diagram, block definition diagram, Internal block diagram, parametric diagram, package diagram, activity diagram, sequence diagram, State machine diagram and use case diagram.

In this thesis three diagrams are used to design requirements of the test case. These diagrams are Requirement diagram, use case diagram, and block definition diagram. A requirement diagram is a diagram which shows a group of requirements and relations between them. The purpose of the requirement diagram is to specify functional and nonfunctional requirements of a system.

2.3 IEC 61499

It is a standard for automation, which is based on the IEC 61131 standard. It provides a component solution for industrial distributed automation systems. It gives added benefits of portability, reusability, interoperability, reconfiguration of distributed applications. In short it provides a generic model for distributed systems. This model includes processes and communication networks as an environment for embedded devices, resources and applications [7]. There is an important concept of Function blocks in IEC 61499, which says that a distributed application is made of a group of function blocks. It is a basic model, which provides interfaces for event and data input outputs. Function blocks are generally of two types:

- Basic Function block
- Composite function block

As the name suggests, a composite function block is composed of basic function blocks and/or other composite function blocks thereby helping in designing a module.

The predecessor of IEC 61499 is IEC 61131. And it is an improvement over its predecessor such as the cyclic execution model that was replaced by the event driven execution model and it results in explicit specification of the execution order of function blocks.

IEC 61499 does work on algorithms and for that purpose they adopted some programming languages from IEC 61131-3 such as Structured Text, Instruction List, Function Block Diagram, and Ladder Diagram [22]. And they also support java too for this purpose. For understanding the program logic inside a simple FB an additional graphical model is used since algorithms are treated differently from rest of the FB models.

2.4 Approaches for control systems

A control system is a system that provides a controlled output by varying the input parameter. It can be of SISO (single input single output system) or MIMO (multiple input multiple output) type. They can be of type open loop or closed loop based on feedback path. CPPS are distributed control systems since it's a very big system and signals are coming from all sorts of sub components/subsystems and every subsystem will need to communicate with each other in a controlled way on various layers of the system.

Control systems make a comparison of a value of process variable, that is being controlled with a desired value setpoint and take the difference and apply it as a control signal to bring the output process variable to the same value as the setpoint. A feedback mechanism is very crucial in control systems to have more control over the process or operation.

2.5 Discuss the UML/SysML – EIC integration

UML stands for Unified modelling language and SysML stands for system modelling language. UML is a standard language for software or system modelling so that it can be visualized and documented before development even starts. Software design and best practices are implemented while designing the software systems. The high-level architecture of a software can be designed by UML. There are many diagrams which represent UML diagrams like Use Case diagram, Class diagram, package diagram, component diagram, deployment diagram, composite structure diagram, profile diagram etc. These are called UML 1.0. As everything in the world, even UML went for a change to version 2.0, which is called UML 2.0. This UML 2.0 coincides with SysML to some extent. SysML is the language for system designers which comes handy for designing the systems. As for example if we are going to design a Car(system), we have to take many aspects into account, for example, functional part, performance part, Structural part and other parts etc. And every part needs to be modelled to have a proper understanding of the system. The benefits of model-based system engineering is the common understanding of system requirements and lead to validation of requirements and identification of risks. Complex system development through incremental development, separation of views through multiple views of integrated model and quality design and enhanced knowledge capture are some of the benefits of SysML. SysML is different from UML in the respect that SysML contains some UML features and some of its own features like bdd (block definition diagram) and ibd (Internal block diagram) etc.

IEC61499 is the function block model defined by The International Electrotechnical Commission for tackling the challenges in automation systems. The major use of this is to model the software systems. For example, in particular in an integrated development environment involving software and mechanics as in cyber physical production systems, the software part can be modelled using IEC61499, while the mechanics part can be modelled using other tools such as modelica, etc. SysML gives the main view of any complicated systems but domain specific views are also taken into account. As IEC61499 gives the software part view and modelica or other tools gives mechanics view of the system. IEC61499 and Modelica come under domain specific languages which are used to carry out the analysis of the model and its execution too. Some diagrams that come under SysML are bdd, ibd. A block definition diagram is used to specify the properties of the system while internal block diagram is used to specify the architecture that the system designer proposes to meet the system level requirements [25]. For a complex mechatronic system defined in [25], the SysML requirement diagrams are best suited to extract mechatronic systems (MTSs) level requirements both functional and non-functional. UML Use case diagrams can be used to extract the system level functional requirements through essential use cases. These use cases are defined in an easy to understand language signifying users' intentions. The system responsibilities or workings are defined with use-case diagrams and picture becomes more clear and then what actors have what responsibilities are defined with activity diagram for specific use cases of the system. The list of activities that need to be performed gets listed in an abstract way. The definition at this stage is abstract functions performed by the system but not its actual imple-

mentation. Other way is that we can simplify the use cases into more simpler terms by breaking it down to sub-use cases to have more modularity in defining system requirement artifacts. It allows us to have more re-use of software components for a real-world application based on requirement specifications defined with respect to different use cases [25]. Both scenarios allow us to have a high-level abstraction view of complex systems with properly defined boundaries. After this stage we can have SysML diagrams like bdd (block definition diagram) coming handy to define the different system properties and ibd (internal block diagram) to define the internal system architecture of the blocks to fulfill the requirements of a system that the system designer proposes. From the paper [25], a bdd diagram that is synthesized in visual paradigm tool to represent first level decomposition of the entire mechatronics system is shown below.

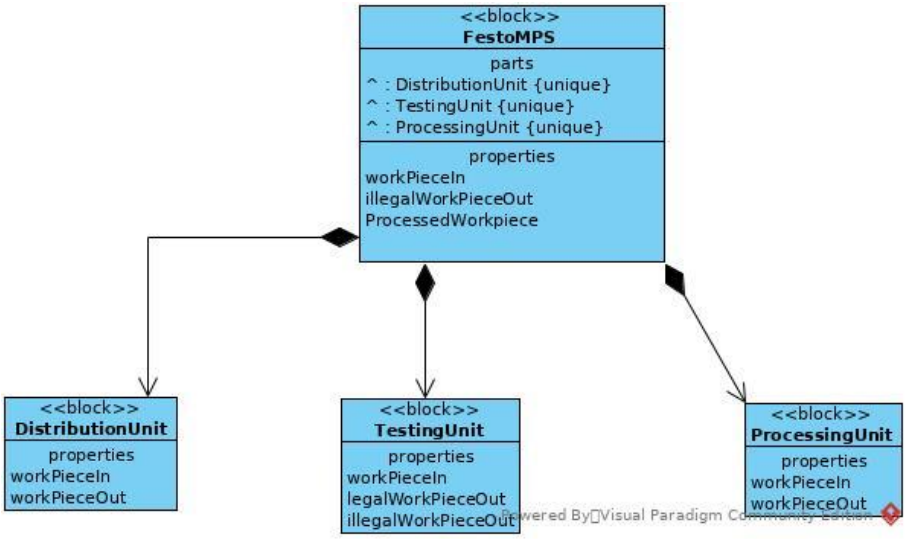

Figure 2.2: Basic capabilities of a CPS and the analogy to CPPS [25]

The internal block diagram designed in visual paradigm is defined below:

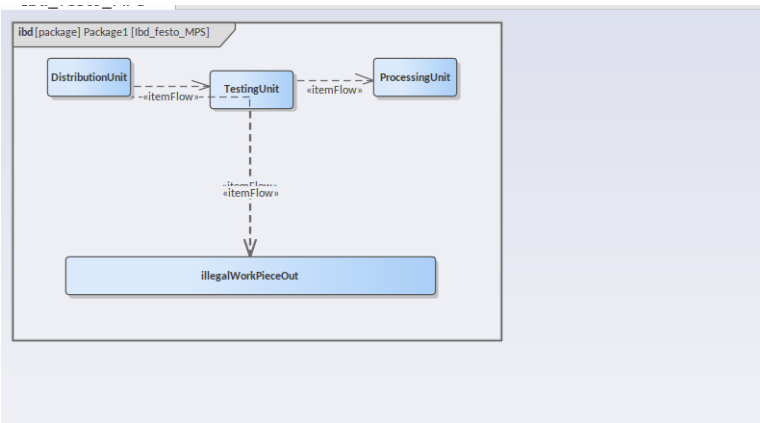

Figure 2.3: Internal block diagram of Mechatronics system [25]

With these steps we are finalized with the V-model of the mechatronics system mentioned in [25]. Thus, each mechatronics component like distribution unit, testing unit and processing unit has well defined responsibilities assigned that are taken care in the following stages. This responsibility assignment in SysML is an effective approach to record each components requirement of the system and allows us easy navigation between different diagrams or models representing or recording these by establishing cross-cutting relationship among them [25]. Two architectures that are popularly used for the definition of architecture are: 1) Bottom-up approach, also known as synthesis and 2) top-down approach, also known as decomposition. Since we explained the bottom-up approach above, for the mechatronics systems some off the shelf components are identified for important system level responsibilities. And an important parameter quality of service (QoS) is used to measure their performance. So, the different proposed components such as distribution, testing and processing components with not too much better Qos are replaced by those with better performance. Other corrective approaches include reengineering of the above collaboration scenario between distribution, testing and processing components. Or a combination of above two approaches both can be used. Alternatively, we can use top-down approach explained in great depth here [24]. Next iteration of the mechatronics requirement is architecting all the mechatronics components. This iteration will end when primitive MTCs were identified such as feeder MTCs. For all the primitive's components identified, the above-mentioned iteration at mechatronics level is skipped and a synergistic integration of three constituent parts of the system that is mechanic, electronic and software is implemented [25]. It is at this point that the feeder MTCs software part is modelled with a function block IEC61499. The mechanical parts modelling is being done by a tool called Modelica.

The SysML model of the system will automatically generate the modelling of the mechanical part of the system in modelica [25]. Authors have explained in [20] how to merge SysML and Modelica constructs for supporting Model-Based Systems Engineering [25]. The draft form of SysML-Modelica integration specification was released by a group working under OMG (Object Model Group) to explain this integration [1]. SysML does not provide support for modeling continuous system dynamics using differential algebraic equations (DAE's) and there is a paper [19] that provides a formal approach for the modeling of continuous system dynamics using DAE's to enable its integration with SysML [25]. The above work gives the foundation as to how we can import automatically the SysML view of the model of the mechanical part in the Modelica tool but the complete integration was not done.

With SysML currently there is no way to execute the models to check the dynamic behavior of the system. The proposed framework in [25] tried to address this challenge by having different modelling tools for mechanical and software part such as modelica for mechanical part and IEC61499 function block for the software part. These provides the base at the system level required for application of MDD (model driven development). The possible tools identified for the translation purpose from UML to IEC61499 are: CORFU-FBDK Engineering Support System, Function Block Development Kit and Rational ROSE.

Chapter 3

Problem statement

In this chapter the problem statement is given.

How to specify a cyber physical production system for a small and medium industries and design a requirement engineering process.

Cyber physical production systems are dynamic control systems with varying degrees of requirements variations at each step. The requirement engineering approach for large systems is already there but for small and medium systems is not defined yet. There is a lot to learn from already developed systems of complex nature so we can derive some approaches from them for our case. In this thesis I am trying to define the approach for the systems on a smaller and medium level so that small and medium industries till now working with traditional systems can also adapt to cyber physical production systems (CPPS) and enjoy its privileges and having value added services and maximum productivity for them.

Till now we have seen the implementation of CPPS/CPS in Industry 4.0, smart factories and in many other places but their implementation is very complex as it involves the interdisciplinary amalgamation of approaches and need all the disciplines to work together in tandem to extract the full benefits.

In section 3.1 some of the challenges of requirement gathering in such systems are defined in detail.

3.1 Requirement Engineering and challenges in the context of CPS

System engineering tools and methods are used for the development of production systems. As it guides the development of complex solutions and production systems has a large number of components that work together for getting the final results. The final developed systems have to be apt and proper and cost effective too and that's why understanding the requirements of the applications and different stakeholders involved becomes very important for successfully engineering the systems. Requirement engineering are needed for planning and initiating the development

process, assessing and testing the impact of changes and of outcomes. If the requirement engineering is not done adequately then it is a source of a whole lot of mess including project failure, exceeding the budgets, missing functionalities and scrapping of projects. System engineering tells that requirement engineering does not only happen in the beginning of development but the interaction between development phase and requirement engineering continues. Same goes for CPPS systems too. A simple production system and a CPS enabled production system are completely different and thus CPS throws a new set of challenges in the requirement engineering process. Due to high complexity of the CPS, many times requirements are fragmented, unrecognized, conflicting, unstable or not fully defined among different disciplines involved. And since the CPS gives cumulative results and results from cumulative interaction of different disciplines, requirement engineering methods and tools must be able to fix issues in requirements and give the predicted results. One other challenge is the communication of different disciplines for effective working of CPS.

Some of the major challenges are defined below.

- Since CPPS systems have an increasing number of tangible and intangible components and rising number of stakeholders from multiple disciplines and thus only that approach that can deal with such complexity and continuously evolutionary nature be suitable to use.
- CPS has integration of hardware and software components and different disciplines are involved in their full life cycle.
- Ensuring consistency of requirement due to the involvement of multiple stakeholders is a challenge.
- Different disciplines need to talk with each other.
- CPS has to consider dynamic user needs in global context.
- With respect to final application scope and focus of requirements change continuously.
- Environment of CPS changes.
- Dynamic system borders of different systems.
- Partially or fully self-governed in real time.

3.2 CPS frameworks for complex systems

There are some popular software and system engineering frameworks for complex systems such as MODAF, TOGAF and SEBok.

These are defined in the sections below.

3.2.1 MODAF framework

MODAF (Ministry of Defence Architecture Framework) and TOGAF (The Open Group Architecture Framework) are two architectural frameworks for developing complex enterprise software systems. MODAF is particularly used for defence planning and change management activities [9]. Change Management activities include implementing change, controlling change and assisting in smooth adaptation of change. It collects and displays the information in a meticulous and a more complete way and helps in comprehending complex issues. This framework can be used by enterprise architects, Architectural modelers, tool developers, engineers, users etc. for serving different purposes.

MODAF has a set of rules and templates which upon filling gives visualization of the concerned business area that is being investigated. This template is called as views. Each view gives different business perspectives to serve all the stakeholders. Different kinds of views are Strategic views (StVs), Operational views (Ovs), Service Oriented views (SOVs), System views (SVs), Acquisition views (AcVs), Technical views (TVs) and All views (AVs). These views are logically and consistently associated by a model known as MODAF which defines the relationship. between all the data in all the views [9]. Some of the advantages of using MODAF are:

- Rigor in requirement capture.
- Modelling of options.
- Interoperability.
- MODAF serves as a base for other advanced frameworks.
- MODAF adopted outside ministry of defence

The below diagram is taken from [10], which properly depicts the MODAF framework philosophy. It has a concept of Views which represent different information in a modelling concept. These views are Strategic views, Operational views, System views, Acquisition views, Technical standard views and All views.

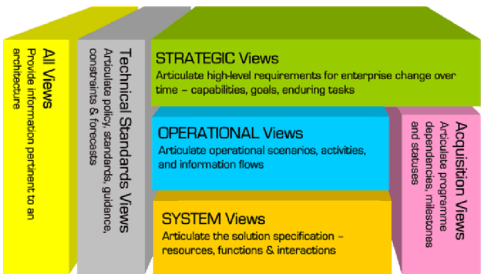

Figure 3.1: Overview of MODAF framework [10]

3.2.2 TOGAF framework

TOGAF stands for The Open Group Architecture Framework. It is a framework for designing, planning, implementing and governing a complex software architecture [11]. TOGAF gives a very high-level design objective. The TOGAF model has 4 levels. These are Business, Application, Data, and Technology. The TOGAF has many features like it has a modular structure. It helps in greater usability, incremental adoption and a library for reference. It has a content framework. This framework helps in having consistent outputs while implementing architectural development methods also known as ADM. It provides a detailed model of the concerned products.

3.2.3 SEBok

For developing a CPPS system, we need to understand system engineering and its best practices that are implemented in development. There is a guide to follow for system engineering and that is called SEBok (Guide to the systems engineering Body of Knowledge). It contains the refined system engineering knowledge and references for consulting for a range of users. This knowledge source is not outdated and comes with frequent updates from community members regarding the latest practices of system engineering. System engineering in an interdisciplinary field of engineering that deals with designing and managing complex systems over their lifecycle. Earlier it was used only for product development but it has evolved to be used in services and enterprises systems. Since system engineering works with different fields so it enables competencies and structure at individual, team and organizational levels. The SEBok is divided into 7 parts which serves as its core principles. These are :

- Introduction
- SE (Software engineering) foundations
- SE and Management
- Application of SE
- Enabling SE
- Related disciplines
- Examples

The part 1 in SEBok gives an overview of SEBok, its role and value in software engineering. Part 2 deals with the knowledge of systems and its role and relation with software engineering. Part 3 deals with standard life cycle processes and best practices. Part 4 deals with the different scenarios and contexts in which these life cycle processes can be applied. Part 5 deals with training individuals, teams or enterprises to adopt good software engineering practices. Part 6 explores other related disciplines involved in life cycle and how we manage to work with them. Part 7 gives examples and lots of examples.

All these frameworks cannot be used in the present context of small CPS systems due to the sheer complexity in requirement capturing from all these frameworks and mostly because those frameworks are specific to large complex systems. Although we can implement these frameworks for small and medium enterprises it is going to be harder due to the small-scale nature of the systems. Presence of no specific methodology for such industries makes it difficult to implement such systems. Having the presence of such framework will make the adoption step easier for industries.

3.3 Conclusion

For complex industrial systems of large magnitude, system specification and thus development is carried with the help of the above defined frameworks like MODAF, TOGAF, SEBok as is done in the case of large scale defence systems and in the case of CPPS systems. These frameworks are as such in their current format not very helpful for small scale systems with small production lines due to the presence of less complexity and less variables involved in system implementation but at the same time need to be adapted to CPPS form. Thus, there is a need for this methodology derived from all these existing frameworks by suitably scaling it down and considering the factors which impact more small-scale systems. In chapter 4, the methodology is developed and in chapter 5 it is validated against a production system.

Chapter 4

Approach

In this chapter we will define the approaches that we will use for gathering the requirements. Section 4.1 defines the methodology and processes for requirement engineering in details. The below diagram aptly describes things in detail.

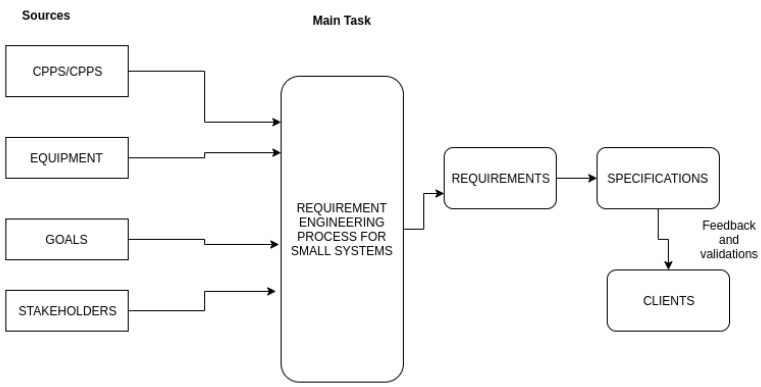

Figure 4.1: Requirement engineering approach

Requirement engineering is the most important phase for the development of such systems. For creating requirement engineering specifications, we will be gathering requirements from UML use case diagrams, SysML block diagrams, SysML requirement diagrams and other techniques of requirement gathering such as brainstorming, interviews, questionnaire etc.

All these approaches and techniques we are employing are for small scale systems as there is no specific requirement engineering approach available for CPPS purpose. With these approaches in place the development of such systems can be pretty easy compared to complex CPPS systems.

4.1 Define the methodology and process:

The methodology involves following steps:

- Gathering of requirements
- Create the model using SysML diagrams
- Map into function blocks (IEC-61499)
- Deploy the system

These steps are explained in the following sections in detail.

4.1.1 Gathering of requirements:

In requirement engineering, requirement elicitation is the process of researching, discovering and gathering requirements of a system from customers, users and different stakeholders involved. The most common requirement elicitation technique/tools include the following as explained in [12] :

- Brainstorming
- Interviews
- Document analysis
- Surveys or questionnaire
- Workshop
- Prototyping
- Observation

Brainstorming creates a list of ideas using words or pictures. Interviews can be in any form formal, informal, in person or virtual. Document analysis is the review of any relevant documentation, schematics or any learning from other previous similar projects. List of questionnaires or surveys with some open ended or close ended questions can be helpful in gathering the right kind of information. Workshop is a tool when we need information from a group of targeted consumers that share similar end user characteristics or a group of SMEs (subject matter expert). Prototyping comes into play when we can make a small version of the product that we are planning to build. Observation can be a tool by observing the stakeholders using prototype or existing mockup and then asking clarifying questions.

4.1.2 Create the model using SysML diagrams:

A model is a small-scale representation of an artifact with abstraction to deal with complexity and is a medium to communicate with stakeholders. SysML stands for system modelling language and is used by system designers to model the system of a product. This phase will include creating SysML/UML diagrams starting from SysML requirement diagrams, which is most suited for extracting system level requirements both functional and nonfunctional of complex systems. UML Use case diagrams can be used to extract the system level functional requirements through essential use cases. These use cases are defined in an easy to understand language signifying users' intentions. The system responsibilities or workings are defined with use-case diagrams and picture becomes more clear and then what actors have what responsibilities are defined with activity diagram for specific use cases of the system. The list of activities that need to be performed gets listed in an abstract way. The definition at this stage is abstract functions performed by the system but not its actual implementation. Another way is that we can simplify the use cases into more simpler terms by breaking it down to sub-use cases to have more modularity in defining system requirement artifacts. It allows us to have more re-use of software components for a real-world application based on requirement specifications defined with respect to different use cases [1]. Both scenarios allow us to have a high-level abstraction view of complex systems with properly defined boundaries. After this stage we can have SysML diagrams like bdd (block definition diagram) coming handy to define the different system properties and ibd (internal block diagram) to define the internal system architecture of the blocks to fulfill the requirements of a system that the system designer proposes. Once these models are defined, we can have requirement specifications out of it as we can filter out the requirements that we need. The modelling diagram for the pizza production system is designed in visual paradigm to capture requirements.

4.1.3 Map into function blocks:

SysML gives the main view of any complicated systems but domain specific views are also taken into account. The mechanics part can be modelled in a tool called modelica for example, while the software part can be modelled using the concept of IEC61499 in other tools. The concept of IEC61499 is based on function blocks, which are reusable software components and robust for complex automation and industrial control systems, best suited for CPPS. The IEC61499 is the industry standard for complex systems having very high integration and performance. There is an important concept of Function blocks in IEC 61499, which says that a distributed application is made of a group of function blocks. It is a basic model, which provides interfaces for event and data input outputs. Function blocks are generally of two types:

Basic Function block and Composite function block

As the name suggests, a composite function block is composed of basic function blocks and/or other composite function blocks thereby helping in designing a module.

UML artifacts can be translated into FB (Function block) environments based on 2 approaches as explained in [22].

First approach includes the usage of UML class diagrams for translation purposes. Using the tool Rational ROSE this transformation can be done. For the hierarchy of function blocks, they are aggregated through aggregation relationships in the UML-FB diagram. And the intra-level and/or inter-level connection between FBs are displayed by association relationships. Each class diagram has 2 important aspects: attributes and operations and these are mapped to function block variables and function block events.

The resulting XML representation of the FB system displays these transformations.

In the second approach, the CORFU-FBDK Engineering Support System is used to depict a model model-driven approach for integrating UML notation into the FB environment as proposed in [23].

The tools that are identified in these two approaches are: CORFU-FBDK Engineering Support System, Function Block Development Kit, Rational ROSE and Function block development kit.

Since for the work we are representing here, we need the knowledge of IEC 61499 as it is the function block of control and distributed systems and CPPS systems are mostly distributed control systems itself.

4.1.4 Deploy the system

4DIAC tool is one of the tools that can be employed for implementing and deploying configuration pipelines into network devices within a CPPS system. It is particularly a graphical user interface (GUI) tool which eases the design and deployment of function blocks. As explained in [4], 4diac is provided as an open source software and it has 2 parts: One is forte and other is 4diac-ide. Forte supports IEC61499 and is a real time run time supporting many hardware platforms. And 4diac-ide is an IDE (integrated development environment) for IEC61499 compliant programs.

4.2 Summary of the proposed approach

The requirement engineering process is summarized in the flow of the process by displaying different SysML diagrams that are used to create specifications of requirements and then mapping into function blocks. All the diagrams such as UML/SysML use case diagram, SysML block diagrams, requirement diagrams and other methods of capturing information's such as conducting interviews, questionnaire, brainstorming etc. played a crucial role in capturing and understanding different intricate details and ultimately requirements of the system by creating various models. The resulting requirement specification is further needed to be translated into function block with the help of approaches defined in section 4.1.3 and finally deployed to the CPPS system with a tool called 4DIAC.

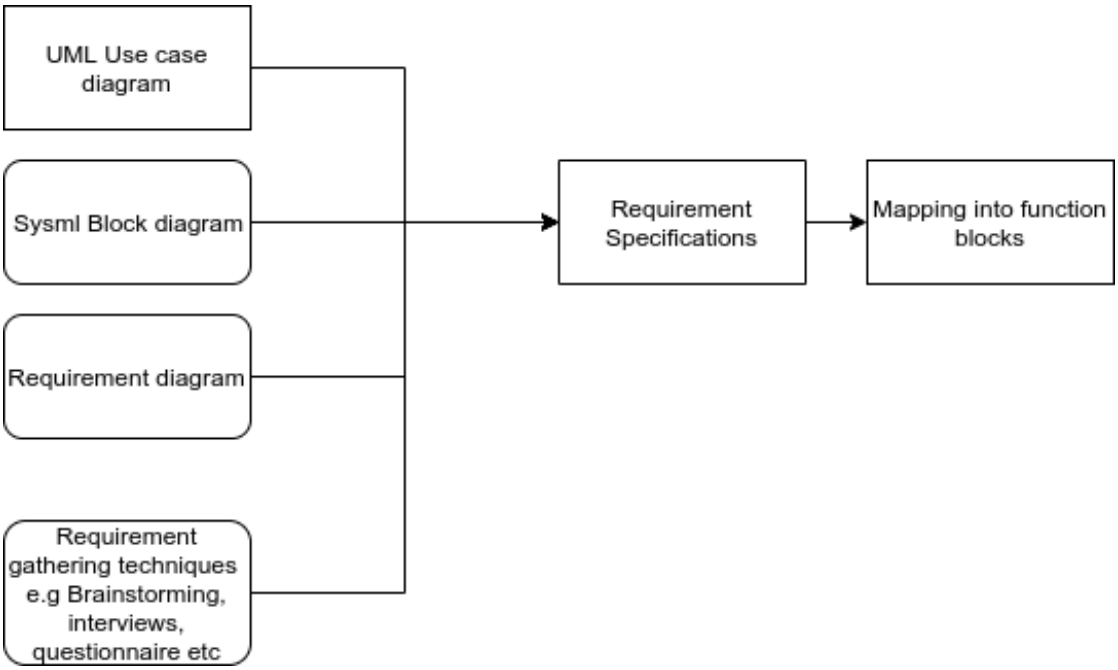

Figure 4.2: Detailed Requirement engineering approach

Chapter 5

Validation of the approach

In this chapter we will validate the requirement engineering approach on a synthetic test case and implement the approach in detail.

In section 5.1 an automated production system as a synthetic test case is defined and all the SysML and UML tools are used to gather requirement with various SysML and UML diagrams such as use case diagram, block diagram, requirement diagram for extracting requirements and then later use case tables are defined to define each use case and then requirements are extracted based on requirement diagram of a mid-level complex systems.

5.1 The automatic production system

This is a synthetic test case on which we are going to apply the methodology that we will implement in the final system. It will showcase how we are going to implement a CPPS system to a test case.

The Digital and Intelligent Industry Lab (DIG2)1, a research laboratory from FEUP, is developing an Industrial IoT platform that will be used as a testbed to demonstrate results from the research projects being developed in the lab.

The main objective is to use this production line example to demonstrate and exemplify various technologies used in the industrial environment in a simple and fun way. An Industrial IoT platform is all about connected assets - machines, equipment and sensors - using various protocols and collecting assorted data and applying control algorithms. The planned platform will implement different use cases, from (remote) condition monitoring and predictive maintenance to analytic frameworks and process control loops. The IIoT platform will be developed having in the background the idea of an autonomous pizza production line. The production line consists of robotic arms, conveyor belts or AGVs (Automated guided vehicles), sensors and pizzas. The pizzas (products) and the ingredients (components) to be used to produce the pizzas will be 3D printed parts.

The test bed will be developed as a toy problem, instead of specifying a real industrial scenario. In this case, we want to develop a generic (but complete) production line, mainly built with 3D printable components, such as 3D printable robotic arms as shown in [2] and AGVs as shown in [8]. The controllers are based on raspberry pi.

The IIoT will be connected to an information system, which will mimic the function of a Manufacturing Execution System (MES) for production order input and production line layout management.

Each robotic arm is a workstation, where a specific production task occurs, and the AGVs are used not only as transport elements but also placeholders for the operations done by the robotic arm.

The layout of the testbed should be re-configurable and the number of elements (robots and manipulators) adjusted. One possible layout is the one presented in the following picture (taken from [27])

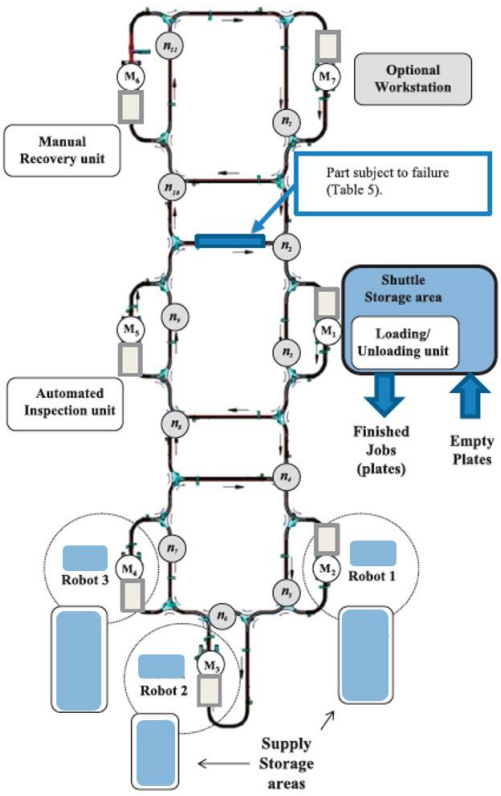

Figure 5.1: Pizza production system diagram [27]

This testbed is connected to a Manufacturing Execution System (MES) which is responsible for sending and monitoring production order, collecting and storing information, presenting and displaying KPIs (dashboards).

Remote access to the testbed is required, both at the MES level and at the control level; control architecture and modules can be developed and deployed remotely.

Access to the testbed must be controlled, with security and access control mechanisms implemented.

5.2 Application of requirement engineering approach

For this system to first develop we need to start with requirement elicitation and starting from all the tools and techniques that we had for requirement elicitation, we are going to make use of the major techniques such as: Starting from talking to customer that what exactly they want and what is the purpose of each component. We will make use of brainstorming to create a list of ideas using words or pictures to create the pieces that will make up the whole system. Interviews will also be done in any form formal, informal, in person or virtual to get more insight. List of questionnaires or surveys with some open ended or close ended questions can be done also in gathering the right kind of information. Studying any relevant document, schematics or any learning from other previous similar projects is done to analyze and review the requirements that can be gathered out of it.

An interview was conducted to gather requirements and it is embedded in the Annexes section.

Following the above process, we understand that the target system needs to do following task at least or have the following requirement:

- It should be automated and produce pizza
- It should have a system for mixing ingredients
- It should have automated inspection unit
- It should have a manual recovery unit in case system fails
- It should have an optional workstation.
- It should have a loading/unloading unit to give out finished jobs(plates) and take back empty plates inside.
- It should have a supply storage area.

For all these basic requirements to take it forward, we need to make use of SysML/UML diagrams to model it and study the model to extract more information. The SysML requirement diagrams are best suited to extract mechatronic systems (MTSs) level requirements both functional and non-functional. UML Use case diagrams can be used to extract the system level functional requirements through essential use cases. These use cases are defined in an easily understandable language showing users intentions. The system responsibilities or workings are defined with use-case diagrams and picture becomes more clear and then what actors have what responsibilities are defined with activity diagram for specific use cases of the system. The list of activities that need to be performed gets listed in an abstract way. The definition at this stage is abstract functions performed by the system but not its actual implementation. Other way is that we can simplify the

use cases into more simpler terms by breaking it down to sub-use cases to have more modularity in defining system requirement artifacts. It allows us to have more re-use of software components for a real-world application based on requirement specifications defined with respect to different use cases [1]. Both scenarios allow us to have a high-level abstraction view of complex systems with properly defined boundaries. After this stage we can have SysML diagrams like bdd (block definition diagram) coming handy to define the different system properties and ibd (internal block diagram) to define the internal system architecture of the blocks to fulfill the requirements of a system that the system designer proposes.

The synthetic test case is modelled so as to capture various requirements with various SysML and UML diagrams. For the purpose of this synthetic test case we have modelled in visual paradigm tool the use case, block definition, internal block definition and requirement diagrams to capture various requirements.

5.2.1 Use case related Diagrams

This section will comprise of the various use cases that are captured while analyzing the system and specifying the major and minor use cases. This section has a use case diagram and use case tables explained in detail in the subsection below to deal with all the aspects of use cases.

5.2.1.1 Use case Diagram

The purpose of a use case diagram is to give a high-level view of the systems flow of events and responsibilities and tasks to different stakeholders in a diagrammatic format. It can capture a complete functional and technical view of the systems. The use case diagram for the system is below:

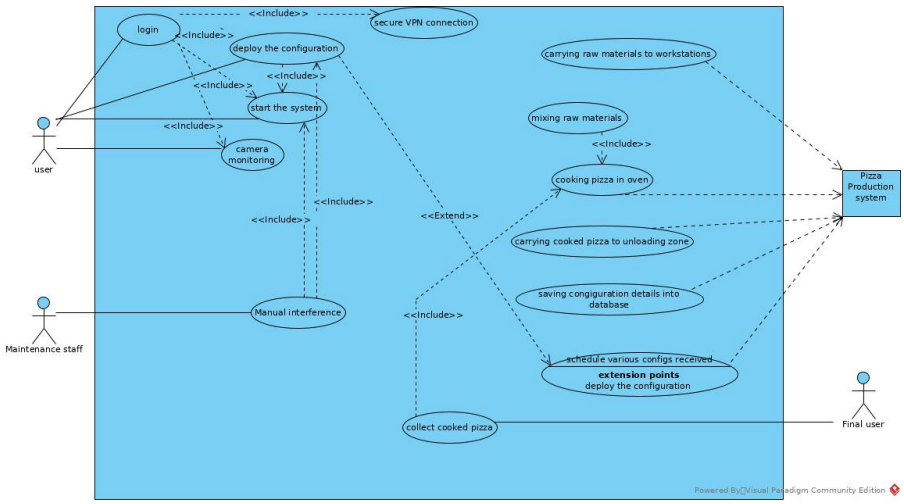

Figure 5.2: Use case diagram of pizza production system

The use case table aggregates all the use cases with preconditions and post conditions for easy analysis of various use cases and possible requirements out of it.

5.2.1.2 Use case Tables

The use case tables groups all the possible use cases with their actors/sub-actors, pre and post conditions and thus flow of events. These use cases are captured from an use case diagram for better understanding of flow of events and eventually gather requirements from them. These tables help in connecting the dots between all the other use cases and thus a complete understanding can be achieved with this. All these diagrams are mentioned below:

Use case	Login
Use case ID	UC1
Actor/supporting actors	Remote user
Pre-condition	The user needs to go to the web URL on a safe network through VPN connection.
Post-condition	The user selects the configuration that needs to deploy and clicks on it.
Flow of events	User goes on a secure VPN connection and clicks the web URL and logs in with its credentials to select the configuration and clicks it.

Table 5.1: Login Use Case table

Use case	Deploy the configuration
Use case ID	UC2
Actor/supporting actors	Remote user
Pre-condition	The user logs into the system.
Post-condition	The CPS based pizza production system receives the signal at receiver and system starts and MES starts configuring the hardware component based on that.
Flow of events	Logged in user deploys the selected configuration and it is received by some receiver at the pizza production system and system starts and MES starts the configuration of system components.

Table 5.2: Deploy the configuration use case table

Use case	Start the system
Use case ID	UC3
Actor/supporting actors	Remote user
Pre-condition	CPS Pizza production system receives the configuration from the system user and starts the system.
Post-condition	The CPS based pizza production system has MES(Manufacturing execution system) that initiates the configuration of system components.
Flow of events	System starts and the configuration of system components starts.

Table 5.3: Start the system use case table

Use case	Camera Monitoring
Use case ID	UC4
Actor/supporting actors	Remote user
Pre-condition	The user needs to log into the system.
Post-condition	The user can see a live feed on a web link all the process that is happening and the product output generated too.
Flow of events	Camera monitoring can display all the products and processes happening live on a web url to supervise the system.

Table 5.4: Camera Monitoring use case table

Use case	Manual Interference
Use case ID	UC5
Actor/supporting actors	Support/Maintenance staff
Pre-condition	If the system develops some snag or needs maintenance.
Post-condition	The system is repaired and fixed.
Flow of events	Manual interference fixes any problem or replaces any components that needs urgent repair.

Table 5.5: Manual Interference use case table

Use case	Carrying raw materials to workstations
Use case ID	UC6
Actor/supporting actors	Pizza production system
Pre-condition	AGVs receive signals from MES and move from the waiting area to get raw materials.
Post-condition	Raw material is carried to workstations.
Flow of events	On receiving signals from MES, AGVs carry the raw material to workstations.

Table 5.6: Carrying raw materials to workstations use case table

Use case	Mixing raw materials
Use case ID	UC7
Actor/supporting actors	Pizza production system
Pre-condition	AGVs carried and placed the raw materials into the workstation platform.
Post-condition	After the raw materials are mixed up it is ready for cooking in the oven.
Flow of events	Mixed raw materials are placed in the oven for cooking purposes for some time.

Table 5.7: Mixing raw materials use case table

Use case	Cooking pizza in oven
Use case ID	UC8
Actor/supporting actors	Pizza production system
Pre-condition	Raw materials are mixed by robotic arms at workstations.
Post-condition	After pizza is cooked then it is placed back to AGV.
Flow of events	Cooking pizza will take some time and then cooked pizza is placed into AGVs.

Table 5.8: Cooking pizza in oven use case table

Use case	Carrying cooked pizza into unloading zone
Use case ID	UC9
Actor/supporting actors	Pizza production system
Pre-condition	Cooked pizza is placed into AGVs.
Post-condition	Cooked pizza is unloaded into the unloading zone.
Flow of events	Cooked pizza is carried by AGVs and unloaded into the unloading zone.

Table 5.9: Carrying cooked pizza into unloading zone use case table

Use case	Saving configuration details into the database.
Use case ID	UC10
Actor/supporting actors	Pizza production system
Pre-condition	The system receives a signal from the user about a particular configuration that needs to be deployed.
Post-condition	Database saves information about the configuration and types of products and number of products produced into the database.
Flow of events	After receiving a signal from the remote user about the type of configuration then the database saves it along with the type of products associated with that configuration and the number of products produced.

Table 5.10: Saving configuration details into the database use case table

Use case	Schedule various configurations received.
Use case ID	UC11
Actor/supporting actors	Pizza production system
Pre-condition	The system receives various configurations from more than one user.
Post-condition	The system schedules the different configurations received and only one configuration is executed at a time.
Flow of events	If more than one configuration then a scheduler puts different configurations in the queue and only one is executed at a time.

Table 5.11: Schedule various configurations received use case table

5.2.2 Block definition diagram

A block definition diagram displays system components, its contents, interfaces and relationship between them. All the system components are drawn as a block and these are static. These blocks can be hardware or software. It supports both required and provided interfaces. And these blocks can be further divided into smaller parts too. A block definition diagram (bdd) can give a lot of information about a system and its components.

Block definition diagram for the test case system is below:

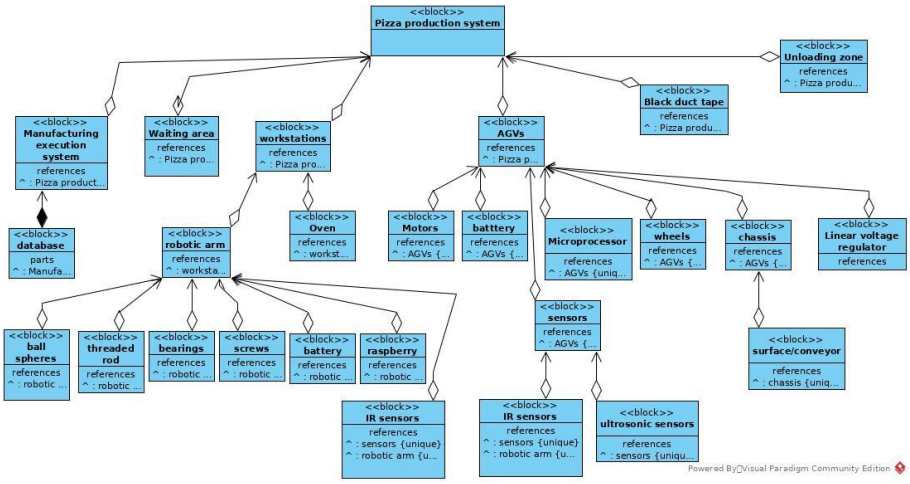

Figure 5.3: Block diagram of Pizza production system

Since the requirement specifications for such a complex system could not be put in a single diagram, it is divided into multiple diagrams:

5.2.3 Requirement related diagrams

All the requirement diagrams and extracted requirements are defined in the form of a table specifying all the aspects of a requirement.

5.2.3.1 Requirement diagram

A requirement diagram is a concept/diagram in SysML which models requirements and relationships between them. A requirement can be a complete or it can be derived also from other requirements too. It helps in extracting and viewing the requirements in the graphical format.

Requirement diagram for the test system is below:

The below diagram displays all the major requirements of the pizza production system. Since it was not possible to design requirements of such a big system in the single window of the tool Visual paradigm, the diagrams are splitted into many sub diagrams to give the complete picture. This below diagram comprises some of the major requirements such as security requirements, system performance requirements, safety requirements, camera monitoring etc.

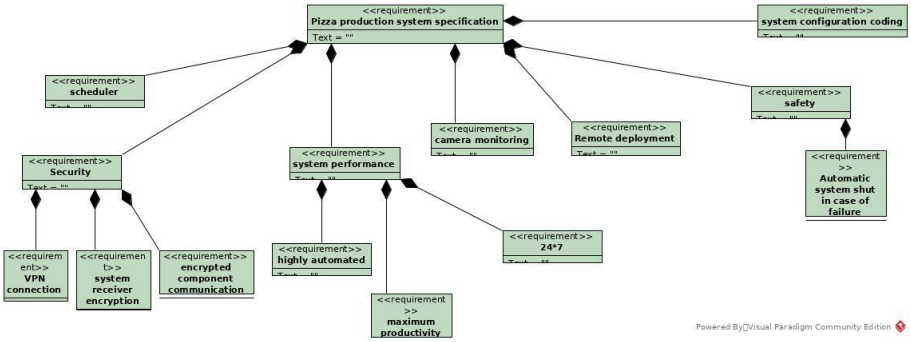

Figure 5.4: Requirement diagram of pizza production system

This below diagram gives more information about its requirements of robotic arms, AGVs, carrying raw materials, cooking pizza etc

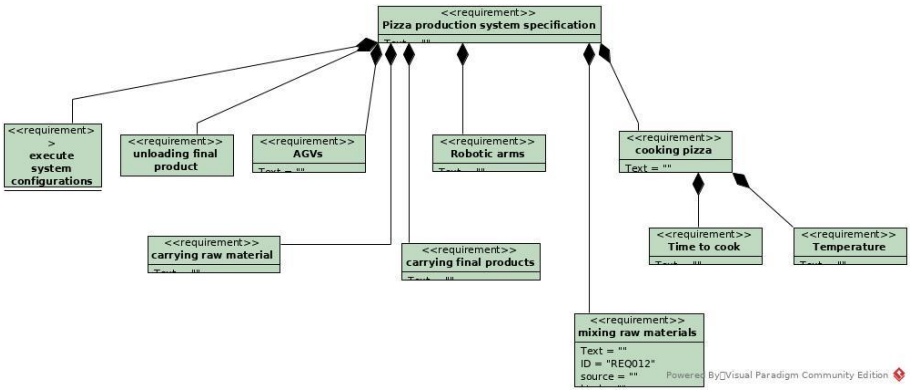

Figure 5.5: Requirement diagram of pizza production system (rest)

AGVs are automated guided vehicles to carry the raw materials into workstations and cooked pizza is carried to the unloading zone. The below diagram gives all the details about AGVs and its associated requirements since it's such a major requirement such as 3D printing, microprocessor, durable batteries etc.

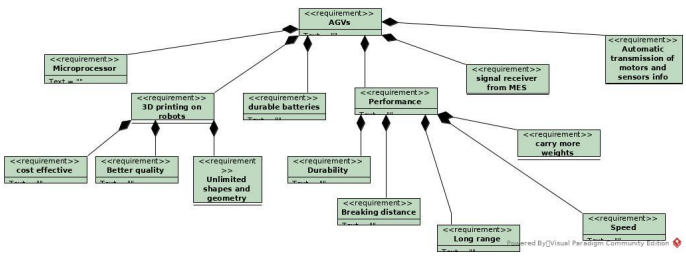

Figure 5.6: Requirement diagram of AGVs (automated guided vehicles)

System configuration coding is the coding of function blocks that ultimately gets deployed to the system. And it is such a big requirement that it is subdivided into sub requirements in the

below diagram.

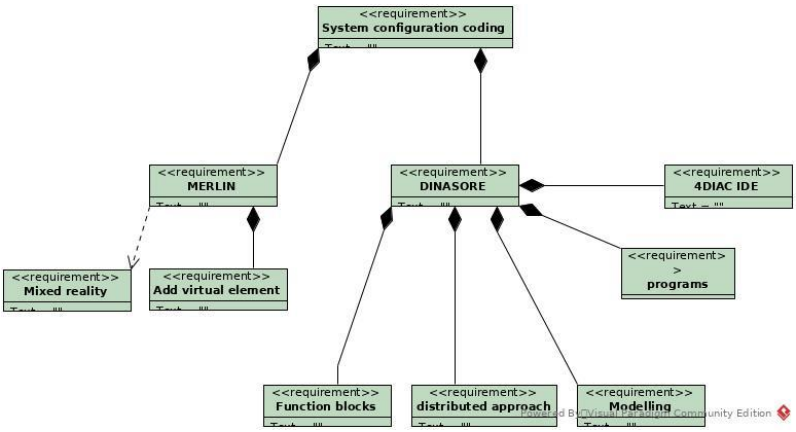

Figure 5.7: Requirement diagram of System configuration coding

Robotic arms is used to mix the raw materials, cook the pizza and put it back to AGVs. Its associated sub requirements are defined below:

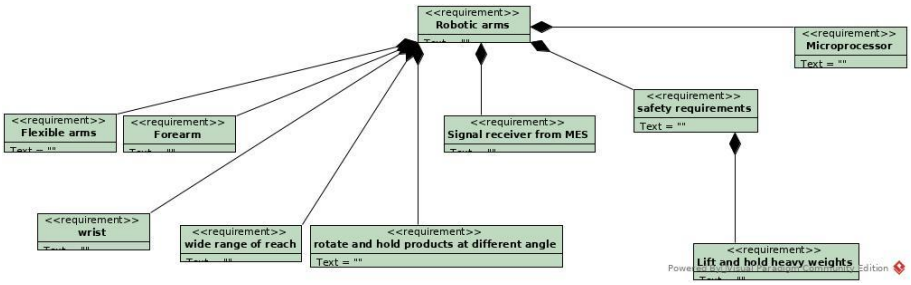

Figure 5.8: Requirement diagram of Robotic arms

From all these requirement diagrams, we can gather a lot of system level requirements that are grouped into some general and some performance and some quality requirements below:

5.2.3.2 Requirements Table

From all these diagrams mentioned above and analyzing them some of the major requirements are grouped in the tabular format below. These requirements are grouped into performance requirements, security, safety and general requirements.

5.2.3.3 Performance Requirements Tables

The performance requirements are listed in below tables:

S.No	Performance Requirements
1	Less Manual interference
2	It works 24 by 7
3	Maximum productivity
4	Less failure
5	Reliability
6	Serviceability
7	Scheduler to schedule multiple configuration request from multiple remote users.
8	The distance at which AGVs put brake to stop
9	The speed of moving AGVs
10	Maximum load and weight that AGVs can carry

Table 5.12: Performance Requirements table

5.2.3.4 Security Requirement Tables

The security requirements are listed in below table:

S.No	Security Requirements
1	VPN connection
2	System Receiver encryption, receives encrypted signals
3	Encrypted communication

Table 5.13: Security Requirements table

5.2.3.5 Safety Requirement Tables

The safety requirements are mentioned below:

S.No	Safety Requirements
1	Automatic system shutdown in case of failures

Table 5.14: Safety Requirements table

5.2.3.6 General Requirement Tables

The general requirements are listed below:

S.No	General Requirements
1	Camera Monitoring
2	Remote deployment.
3	Execute system configurations
4	Unloading final products
5	Need Automated guided vehicles
6	Need Robotic arms
7	Carrying final products
8	Cooking pizza
8.1	Time to cook needs to set.
8.2	Temperature need to set
9	Mixing raw materials
10	Robots be it AGV or Robotic arm needs to be 3D printed
10.1	3D printed robots are cost effective
10.2	3D printed robots are better quality
10.3	3D printed robots support unlimited shapes and geometry as per requirement
11	Automatic transmission of motors and sensors information
12	System configuration coding
12.1	MERLIN app to configure virtual robots and other objects into the environment (for students)
12.2	DINASORE design and implement CPPS, by reconfiguring data workflows and deploying configuration pipelines in actual physical devices in a distributed manner
12.2.1	IEC61499 needed for distributed control systems
12.2.1.1	Function blocks are building blocks of IEC61499
12.2.1.2	Distributed approach is required for CPPS based systems
12.2.1.3	Distributed modelling of the system in which programs will run is needed to simulate actual systems
12.2.2	Programs need to be written in java/python that will be deployed in function blocks to make it functional
12.2.3	4DIAC IDE is the GUI to ease the design and deployment of FBs to reconfigure OnDemand target devices in the CPS.

Table 5.15: General Requirements table

After modelling into various diagrams, now we need to make use of the IEC 61499 function block model to model the software components. A run time environment based on CCM (Corba component model) will be used to implement the packaging, assembling, deploying and executing applications as aggregations of components. It allows the integration of function blocks (FB) types with different language implementations such as C or C++ or Java, etc. or any operating systems or any hardware nodes as described in [26].

The above SysML models extracted requirements from the system but now the mapping into function blocks is done by the help of UML class diagrams. Using the tool Rational ROSE this transformation can be done. For the hierarchy of function blocks, they are aggregated through aggregation relationships in the UML-FB diagram. And the intra-level and/or inter-level connection between FBs are displayed by association relationships. Each class diagram has 2 important aspects: attributes and operations and these are mapped to function block variables and function block events. The resulting XML representation of the FB system displays these transformations.

After the function blocks are designed, we can deploy it with the help of 4DIAC-IDE tool which is primarily a graphical user interface to ease the design and deployment of function blocks to reconfigure on demand target devices in the CPS systems

5.3 Discussion of the result and suggested improvements

The software engineering methodology developed in chapter 4 is validated with a test case of pizza production system with the implementation of all the steps developed in chapter 4. The methodology works well to integrate cyber physical production systems in the test case. We were able to extract requirements by modelling requirement diagrams. Block diagrams gave us the major components of the system and use case diagrams gave us the flow of events and system level interaction. We were able to extract safety requirements, security requirements and general requirements. And the approach of mapping SysML diagram to function blocks was also mentioned and few tools such as CORFU-FBDK Engineering Support System and Function Block Development Kit were also discovered which helps in translating to function blocks. And then the deployment to physical devices with the help of 4DIAC is well explained.

Chapter 6

Conclusions and future work

In this thesis, a requirement engineering approach of cyber physical production systems for small scale industries was developed which would make the adoption among such industries vary high.

Software engineering approaches for large and complex systems already exist but for small scale industries, implementing the same approach is not very straightforward and also not very easy due to less complexity and less variables involved in the working of such industries and the existing approach is very vast. The approach developed was based on system level understanding of such systems at both hardware and software level with the help of SysML diagrams for understanding the requirements before applying the CPPS features.

The requirement engineering approach proposed in the work although depends on a lot of research and research papers but still as every system big or small is different in its own way and type of products produced thus the final implementation of CPPS behavior will be different. CPPS provides a lot of features that we can take advantage of depending on whether we are developing the system from scratch or transforming an already developed system for adoption to CPPS.

SysML Requirement diagram is one of the most useful tools that helps in capturing requirements of systems apart from other UML/SysML diagrams.

For developing the approach, it was extremely crucial to understand the system at the fundamental level and what tools to use for that at each level. Studying different papers and bibliographies to gather information about implementation of CPPS was extremely difficult as right information was scattered and too much elusive. Developing a strategy to map UML/SysML diagrams into IEC61499 function blocks was the most crucial step for deploying function blocks into physical systems. For a complete software engineering of a CPPS system right from understanding the system to extracting requirements to deployment is mentioned in the thesis.

CPPS, popularly called the 21st century production system has many advantages and that's why its adoption among every section of industry is very important and it has the potential to enhance the output as well as provide high level of efficiency due to the complete automation and many benefits that it offers.

The result of this thesis helps in understanding the CPPS system in context of small-scale systems specifically and development flow from requirement gathering to deployment also for small scale systems.

The future work involves more research to make the requirement engineering process as flawless as possible and must consider various scenarios. In this thesis we have tried to make a base for software engineering approach with a use case of a pizza production system that can surely be extended to cover many scenarios in future. The tools that transform UML/SysML diagrams into function blocks are not much advanced and it needs more research and innovation so that the process from UML/SysML to function block can be streamlined. As the CPPS field is continuously evolving so there is a lot of scope to advance data processing and transmission too.

References

[1] Sysml-modelica transformation specification. http://www.omgwiki.org/OMGSysML/lib/exe/fetch.php?media=sysml-modelica:sysml-modelica_xformspec_allparts_3-17-10.pdf, 2009.

[2] 3d printable robotic arm. http://www.eezyrobots.it/eba_mk2.html, February 2020.

[3] 3d printing. https://en.wikipedia.org/wiki/3D_printing, April 2020.

[4] 4diac - framework for distributed industrial automation and control. https://www.researchgate.net/figure/Ministry-of-Defence-architecture-framework-MODAF_fig4_262783461, August 2020.

[5] Cyber physical system. https://en.wikipedia.org/wiki/Cyber-physical_system, March 2020.

[6] Dfki gmbh cyber physical systems. https://www-cps.hb.dfki.de/home, May 2020.

[7] Iec61499 - the new standard in automation. http://www.iec61499.de/, May 2020.

[8] Line follower robot using raspberry pi. https://circuitdigest.com/microcontroller-projects/raspberry-pi-line-follower-robot, February 2020.

[9] Mod architecture framework. https://www.gov.uk/guidance/mod-architecture-framework, May 2020.

[10] Modaf system architecture. https://www.researchgate.net/figure/Ministry-of-Defence-architecture-framework-MODAF_fig4_262783461, August 2020.

[11] The open group architecture framework. https://en.wikipedia.org/wiki/The_Open_Group_Architecture_Framework, May 2020.

[12] Requirement elicitation techniques. https://qracorp.com/requirement-elicitation-techniques/, May 2020.

[13] Smart material. https://en.wikipedia.org/wiki/Smart_material, April 2020.

[14] Sysml open source project. https://sysml.org/, May 2020.

[15] A R Al-Ali, Ragini Gupta, and Ahmad Al Nabulsi. Cyber physical systems role in manufacturing technologies. In *AIP Conference Proceedings*, volume 1957, page 50007, apr 2018.

[16] O. Cardin. Classification of cyber-physical production systems applications: Proposition of an analysis framework. *Comput. Ind.*, 104:11–21, 2019.

[17] José Ferreira, Fábio Lopes, Sudeep Ghimire, Guy Doumeingts, Carlos Agostinho, and Ricardo Jardim-Gonçalves. Cyber-Physical Production Systems to Monitor the Polishing Process of Cutlery Production. In João Martins, Vladimir Jotsov, Robert Bierwolf, João Pedro Mendonça, Ricardo Jardim-Gonçalves, and Maria Marques, editors, *9th International Conference on Intelligent Systems, IS 2018*, pages 926–933 BT – 9th International Conference on Intel. Institute of Electrical and Electronics Engineers Inc., jul 2018.

[18] Volkan Gunes, Steffen Peter, Tony Givargis, and Frank Vahid. A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems. *KSII Transactions on Internet and Information Systems*, 8:4242–4268, dec 2014.

[19] Thomas Johnson, Jonathan Jobe, Christiaan Paredis, and Roger Burkhart. Modeling Continuous System Dynamics in SysML. In *Proceedings of the IMECE 2007*, jan 2007.

[20] Thomas Johnson, Christiaan Paredis, and Roger Burkhart. Integrating Models and Simulations of Continuous Dynamics Into SysML. *Journal of Computing and Information Science in Engineering*, 12, mar 2012.

[21] H.-B. Jun, J.-H. Shin, D. Kiritsis, and P. Xirouchakis. System architecture for closed-loop plm. *International Journal of Computer Integrated Manufacturing*, 20(7):684–698, 2007.

[22] S. Panjaitan and G. Frey. Combination of uml modeling and the iec 61499 function block concept for the development of distributed automation systems. *2006 IEEE Conference on Emerging Technologies and Factory Automation*, pages 766–773, 2006.

[23] Kleanthis Thramboulidis. Using UML in control and automation: A model driven approach. pages 587–593, jul 2004.

[24] Kleanthis Thramboulidis. The 3+1 SysML View-Model in Model Integrated Mechatronics. *JSEA*, 3:109–118, jan 2010.

[25] Kleanthis Thramboulidis and Andrea Buda. 3+1 SysML view model for IEC61499 Function Block control systems. pages 175–180, aug 2010.

[26] Kleanthis Thramboulidis, D Perdikis, and S Kantas. Model driven development of distributed control applications. *The International Journal of Advanced Manufacturing Technology*, 33:233–242, jun 2007.

[27] Damien Trentesaux, Cyrille Pach, Abdelghani Bekrar, Yves Sallez, Therese Bonte, Paulo Leitão, and José Barbosa. Benchmarking flexible job-shop scheduling and control systems. *Control Engineering Practice*, 21:1204–1225, 05 2013.

Appendix A

Interview Questions

The interview was conducted to gather the requirements and questions and answers are listed below.

How the system works currently and how it is expected to work after implementing CPS ?

After implementing CPS, the user will be able to start the process from the internet from any location through a VPN (virtual private network) encrypted network and can supervise the entire system running on a streaming video camera to oversee the entire workstation and waiting station. There are workstations and waiting stations as part of the system apart from MES(Manufacturing execution system). These are computerised systems used in manufacturing to track and document the transformation of raw materials to finished goods. The production line can be divided into two main areas, namely a waiting area for Automated Guided Vehicles (AGVs) and several working stations, each one with a robotic arm. The role of the AGVs is to transport the materials and product increment along the production line, from the waiting zone until the final unloading zone. The surface of the AGV is used also as a working area, for task executing sharing between AGV and robotic arms. Each AGV must follow the black duct tape, which marks the path to travel. Along this path, the AGV stops at the respective working areas, where together with each robot manipulator, a specific product production task is executed. In the end, when the product is produced, the involved AGVs in the process must return to the waiting area for a new production order.

Where is the system expected to start and what are the advantages of it compared to earlier systems ?

This is different from other systems in many ways. The use of a mixed reality environment allows the user to see and interact and to learn a little bit about all these technologies.By using augmented reality the user may understand somehow about the benefits about virtual robots and other objects that can be included in the virtual systems. For remote systems we use DINASORE technology and for more local systems we use mixed reality. DINASORE, is a framework used to design and implement CPPS, by reconfiguring data workflows and deploying configuration pipelines in actual physical devices in a distributed manner.

Since mixed reality is being used in the production system, how do things kick off in the real world and in the virtual world simultaneously ?

This is pretty much like a Pokemon Go for example. We can use our smartphone camera on a component and put on this a virtual object like robotic arms and other objects. Then we can build our entire production line based on this. If we use Mixed reality then we have the functionality to code all of this objects and deploy to a physical setup. For remote deployment we are considering DINASORE which does not have this augmented reality capability. We have a canvas where we drag and drop the function blocks representing application components, which will contain the code that can be deployed on a physical node and reconfigure the node to a given functionality at the end. For industries and professional side we prefer DINASORE.

What business needs the system is expected to address apart from making it autonomous and remote deployment ? What challenges you are facing in building this system ?

Remote deployment is one of the important features where a single entry point will be responsible for deploying the configuration on a network in a distributed manner. It's difficult to understand how it can be done in a remote setup. Regarding the mixed reality one of the issues that we have is we want to give the opportunity to users to use code blocks to create some applications to deploy to physical components. We need to understand how we will interface this codeblocks and this application with the physical environments and how we can do this integration in both. The code blocks has certain limitations on the capabilities that we want them to perform and we need to know how to achieve this.

For remote deployment are we using Raspberry pi or what is the purpose of using this in the system ?

Raspberry pi is used to connect or integrate the physical components of the system so that they can communicate with each other in robot manipulators and AGVs.

Is this system single user or multi user and where the user will be located when deploying the system ?

It is going to be single user system but since people can try to access it at the same time so there is going to be an issue and we will have a scheduler which will schedule the different inputs it receives and configure the system accordingly in case the system receive more than one input or command from different users.

How can we access the system remotely, is it going to be remote control or something and if remote how remote the user can be ?

The user should have an internet connection and a VPN connection to start the process so the user can be miles away and still can access the systems and there will be a live feed coming from a camera monitoring the entire process where we can see the live results.

Does this system depend on any other system for input or does it give output to any other system ?

It does not depend on any other external system for inputs but there is going to be an information system which will mimic the main functionalities of a Manufacturing system (MES) for

production order and production line layout management. This information system will reconfigure the physical components to do a particular task. For this there may be a database where we may store order of the production line and topologies of the product that we want but no information about users.

What input the system should receive and what output the system should produce ?

The input should be a command from a user on a VPN network to start the system and configure the system for a particular product and output should be a pizza or other product which it receives in the input. Moreover the mixed reality or DINOSAUR technology will give us the entry point in the system. In the mixed reality the user will open an app in the smartphone, write some code in the codeblocks and the entry point on the testbed will be theoutput of the codeblocks application. Same way the output of the Dinasaur is the entry point of the testbed.

What are the manual things that are being automated with this kind of CPPS based systems ?

The idea is to automate as many manual tasks as possible, for example getting the ingredients, mixing it together ,putting it in the oven and getting the final output. The only interaction that this system is expected to have is deployment of configuration in the beginning and the system production line will reconfigure accordingly and all the functionalities and execution of tasks will be automated. Only when something goes wrong and during maintenance only human interaction is required.

What are the hardware and software functionalities of the system ?

For triggering the process, what is being done is that when the deployment of the configuration is done then all of the nodes in the system will be reconfigured with the new code which has new functionalities, new parameters and then user can trigger the system to start may be using the information system to create the new production line. As far as intelligence of hardware goes, we can code some advanced function blocks but since the functionalities of all the components are limited to very simple use cases so simple function blocks will suffice. If more functionality is provided then there are chances of something going wrong in the system, for example the robots should only follow a black line while movement but not the free movement.

How much interoperable and scalable is this system ?

As much component we want to add, we can add here, so is highly scalable as we can include more function blocks to all those components on the software part and they will be good to go. For interoperability, this is not an issue since we are using a digital concept where everything is connected by signals or networks so we can connect it to any other third party systems also if it can be virtualised by using the same concept of mixed reality or DINASORE.

What functional limitations that we are trying to change here from the existing systems and what will it bring new from existing systems ?

It is completely automated, removing human interaction to the minimum, remote deployment of the configuration, machines interacting with signals or internal communications and based on different configurations deployed, different products can be the output of the system. This testbed is mainly used for education purposes for students to experiment a little bit in technologies used

in industries. Right now industry is suffering transformation from industrial processes to more advanced processes. This testbed is trying to mimic some of these functionalities.