FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

MODEL BASED SYSTEM ENGINEERING FOR THE DEVELOPMENT OF SYSTEM ON CHIP

Author	Saad Zahoor
Supervisor	Dr. Zaheer Khan
Second Examiner	Dr. Tuomo Hänninen
Technical Advisor	Muhammad Zeeshan Waheed

January 2023

Zahoor S. (2022) Model Based System Engineering (MBSE) for The Development of System on Chip (SoC). University of Oulu, Faculty of Information Technology and Electrical Engineering, Degree Programme in Wireless Communications Engineering. Master's Thesis, 49 p.

ABSTRACT

Model Based System Engineering (MBSE) has been utilized in auto manufacturing industries, airplane manufacturing and maintenance, and factory process automation industries. These are some of the complex fields. As SoC design is a complex process and requires years of work, MBSE can reduce time, complexity, reuse, and maintenance costs. It seems a fruitful idea/decision to take MBSE into use in SoC design depending on the previously mentioned elements. System on Chip (SoC) is obtaining the interest of many big companies. Therefore, MBSE will represent a huge competitive advantage once it is taken fully into the systems engineering roles of SoC. The existence of geographically dispersed teams, complexity of systems, interdisciplinarity, personalized system description, and their integration can be enabled by MBSE. As an emerging paradigm for the systems of the 21st century, MBSE paved the way for creating successful systems (for the companies) that are end to end connected. This research focuses on making use of MBSE in SoC. The thesis will show how SoC processes can be implemented in one complete model with top to bottom approach. Firstly, the traditional systems engineering approach has been explained with its tools and examples. Secondly, the need for taking up MBSE by the systems engineers is expressed. This contains the applications, use in modern systems, and benefits of MBSE. Moreover, MBSE methodology tools, languages, and their use in SoC is illustrated with examples. As SoC development is a huge and complex process; therefore, a small component of the chip has been taken in consideration for the purpose of understanding and making of the thesis. MBSE is a model-based approach hence a language needs to be present to produce these models and that language is SysML and OPD/OPL. SysML language and MagicDraw tool is used for expressing the architecture of the system. MagicDraw supports several external evaluators for evaluation of expressions and MATLAB is one of them. With MagicDraw we can do simulations, input parameters, and analyze data by processing on it using algorithms developed in MATLAB.

Keywords: MBSE, SoC, SysML, MagicDraw, MATLAB

TABLE OF CONTENTS

FOREWORD

The aim of this thesis is to study and demonstrate an approach which will help System on Chip (SoC) engineers to model the chips in a proficient manner. This includes modelling the top to bottom architecture of a subsystem in a modelling language known as Systems Modelling Language (SysML), producing simulation results from MATLAB, and finally merging them to create a combined model to assist SoC engineers. This work is done in collaboration with Nokia Solutions and Networks Oy and the University of Oulu.

Firstly, I would like to express my gratitude to my technical advisor Muhammad Zeeshan Waheed and my supervisor Dr. Zaheer Khan for their guidance and helpful suggestions. Secondly, I want to appreciate Mika Ventola who leads the project at Nokia under whom this thesis was conducted. Thirdly, I want to thank Petro Heikkila for assigning me this thesis. I also want to give special thanks to my colleagues at Nokia, Shufeng Zheng, Marko Kokko, Praveen Buddireddy, and Zhen Ni, who all contributed to making this thesis. Additionally, I would like to express my gratitude to my friends who supported me emotionally and helped me in proofreading the thesis. Finally, I would like to express my deepest gratitude to my parents for supporting me in timely completion of this thesis.

Oulu, January 5, 2023

Saad Zahoor

LIST OF ABBREVIATIONS AND SYMBOLS

AADL	Architecture Analysis and Design Language
ACT	Activity Diagram
ALM	Application Lifecycle Management
API	Access Point Information
ASIC	Application Specific Integrated Circuit
AXI	Advanced eXtensible Interface
BDD	Block Definition Diagram
BPMN	Business Process Modelling Notation
CBS	Computer Based Systems
COSYSMO	Constructive Systems Engineering Cost Model
CP	Cyclic Prefix
dB	Decibells
eMBB	Enhanced Mobile Broadband
EPL	Eclipse Public License
IBD	Internal Block Diagram
IBM	International Business Machine
ICI	Intercarrier Interference
ICs	Integrated Circuits
IFFT	Inverse Fast Fourier Transform
IoT	Internet of Things
IP	Intellectual Property
JUDE	Java and UML Developers' Environment
LTE	Long Term Evolution
MATLAB	Matrix Laboratory
MAV	Mars Ascent Vehicle
MBE	Model Based Engineering
MBSE	Model Based System Engineering
MD	Magic Draw
MDG	Model Driven Generation
MES	Manufacturing Execution System
MRS	Mars Sample Return
NR	New Radio
OFDM	Orthogonal Frequency Division Multiplexing

OPD	Object Process Diagram
OPL	Object Process Language
OSI	Open Systems Interconnection
PA TDD	Phase Adjusted Time Domain Data
PIMs	Platform-Independent Models
PLM	Product Lifecycle Management
RMS	Resource Management System
Rx	Receiver
SE	System Engineering
SoC	System on Chip
SysML	Systems Modelling Language
TDD	Time Domain Data
Tx	Transmitter
UML	Unified Modelling Language
WSN	Wireless Sensor Networks

BER	Bit Error Rate
K	Number of data symbols in the data
SNR	Signal to Noise Ratio

1 MOTIVATION

The work on MBSE methodology is consistent with the work of systems engineer. In fact, MBSE is established as a helping methodology for systems engineering work. It provides a way to make systems in modelled environment that makes it close to the practical solution. System Engineers get specifications from the customer and then convert those to a technical specification. Then the architecture is made on top of that depending on those documents. Using a traditional method, this consumes huge amount of man hours in creating documents and modelling the specifications. These documents and models stay in the system, however; the location of them needs to be found while making the subsystems. MBSE reduces these gaps by defining the architecture in SysML language and storing everything of a project in a single containment tree.

MBSE has been successfully used in automobile industries, airplane manufacturing and maintenance, factory automation industries. Some industries do not go into making phase of the project unless a working model has been developed using MBSE. This makes it extremely demanding for other complex fields such as SoC. Practical solutions of utilizing SysML are available in many companies and every company has their own ways of presenting systems in SysML. SoC development has not been taking it up as much as others. Seeing improvements in other industries efficiency before and after adopting MBSE processes, it is evident that the progress is considerable and SoC should adopt it. This makes it a hot topic that gave the motivation to write this thesis.

2 LITERATURE REVIEW

This chapter provides information about the literature that is being viewed while writing the thesis. It starts with the background of explaining the advantages of executable models. Then SysML adoption, usage, and industrial advantage is described in detail. The traditional systems engineering approach is demonstrated with the tools, and examples. Then comes the need for adoption of MBSE in systems engineering work. Some methodologies currently utilized are explained as well. Additionally, benefits, applications, and a model is expressed for illustrating the idea of adopting MBSE. Moreover, few languages used by systems engineer and tools that use these languages in their work are illustrated. Finally, the environment for the SoC development by using SysML is described.

2.1 Introduction

2.1.1 Background

2.1.1.1 Executable Models Creation Advantage

Executable models in the early stages of the project making are the key drivers in determining the project cost, man hours, and time consumption. The reasons for creating these types of models are, executable models can be utilized for evaluation of crucial choices of early design and eliminate errors. These models can be made for verifying the requirements with interested parties. Additionally, models can be used for inter-model transformation [1].

To communicate with the stakeholders and obtain specific requirements in their respective language is essential for every systems engineer. This is increasing with the passage of time. Now how to communicate in the systems engineer (SEs) language is a key challenge that needs to be addressed. As the technology is changing and evolving continuously in this era it has become mandatory for the industries to make the workflow more efficient. Efficient work force produces best time limited and project specific decisions that help the industries grow.

2.1.1.2 SysML Adoption

SysML is a language that enabled the advent of MBSE methodology in industries. It was created in 2003 for systems engineering applications and was adopted by OMG SysML in 2006. It was awarded in the “Modelling” category in the year 2007 [2]. The current version of OMG SysML is 1.6 [3].

2.1.1.3 SysML Usage

The most common usage mode of SysML is making models from the provided requirements and specifications, hence it is sometimes abused as a mode of generating pretty pictures but there is more to this. The capability of generating automated reports, simulation at several diagrams, conversion to MATLAB/Simulink code possess a huge advantage over the Visio, PowerPoint illustrations [4]. Besides this, these SysML models can be a blueprint of the actual model that contains the whole details of the workflow after the systems engineering requirement work.

2.1.1.4 Industrial Advantage of SysML Adoption

Taking SysML into use also poses a competitive edge in the field of SoC as most of the other industries flourished after adopting SysML [5]. Since most of the work of MBSE is done on factory process automation, car manufacturing industries, and aircraft industries. It appears that there is a little adoption of SysML in SoC or ASIC development currently in any company. Becoming pioneer of it is a great start of any industry to grasp the benefits of SysML and becoming a leader would pose a competitive advantage in setting up the standards for SysML use in SoC.

2.1.2 Project Description Based on Designs

Model development is an iterative incremental process. Designs are the best way to express any description. With designs we can implement all the diagram flows, produce the results of our interest, implement the intent which is helpful in determining the marketing strategy of the product. Visual representations are always preferred when taking marketing decisions as they provide an edge of being easily viewable than the whole bunch of documents.

Explaining the problem visually requires effort and investment into the work. Designs can be logical, functional, and physical models that can also be used when the project is ongoing. Some tools have the capability to do simulation work on these models which makes them extremely useful for the purpose of analysing the impact of the simulated elements before time.

SysML has risen to be the language specifically for the systems engineering work. SE work emphasizes on the making of architecture and specifications in modelling format. Currently these engineers make Visio diagrams, PowerPoint slides to convey the demands. This needs to be addressed, as the technology is evolving quickly, more complex circuits are designed with great precision due to which it becomes obsolete to exhibit a vague view with only the diagrams.

SE needs to adopt the SysML practices that would enable the future development of the products. Model reusability is also a great feature that SysML provides which makes it easy to develop new models in less time. SysML is reducing the effort for the system engineers, hardware engineers, and software engineers as a proper flow can be created and validated between interfaces. This language also provides an advantage of having everything linked to the architecture of a project at a single place that helps in finding the required portion in lesser time.

2.2 The Traditional Systems Engineering (SE) Aspects

As the name suggest the traditional SE aspect is the process of using somewhat old software applications. Traditional name is given here to represent, from top to bottom approach, those industries who are utilizing the concept of explaining the requirements, functional architecture, physical model in a documented manner rather than making models to present these scenarios.

2.2.1 Tools of Traditional SE

The tools that are supporting this traditional theory aspect are:

- Visio diagrams
- Excel sheets
- Google sheets
- Word documents
- Google docs
- Power point presentations
- Visme
- Prezi

These diagrams, sheets, documents no doubt give a good illustration of the idea but the problem lies in going through all these resources and if one or more of the resources is left behind than there is a chance of error in the project creation. Sometimes due to this human error the whole project falls apart.

2.2.2 Example of Traditional SE

We can take an example of it in such a way that, Fig. 01 taken from [6] shows a traditional system engineering process where flows are given. The flows are describing how the project will proceed with the blocks of requirement analysis, functional analysis, synthesis, and system analysis and control (balance). This is the traditional approach of describing the workflow. The internal structure of these blocks cannot be accessed by clicking on them and we need a new illustration to represent their internal structure which may be present somewhere else. This issue is removed by new systems engineering process known as MBSE where we not only create these visual diagrams but also provide some functionality to the blocks to make it easier for the reader to communicate his/her thoughts.

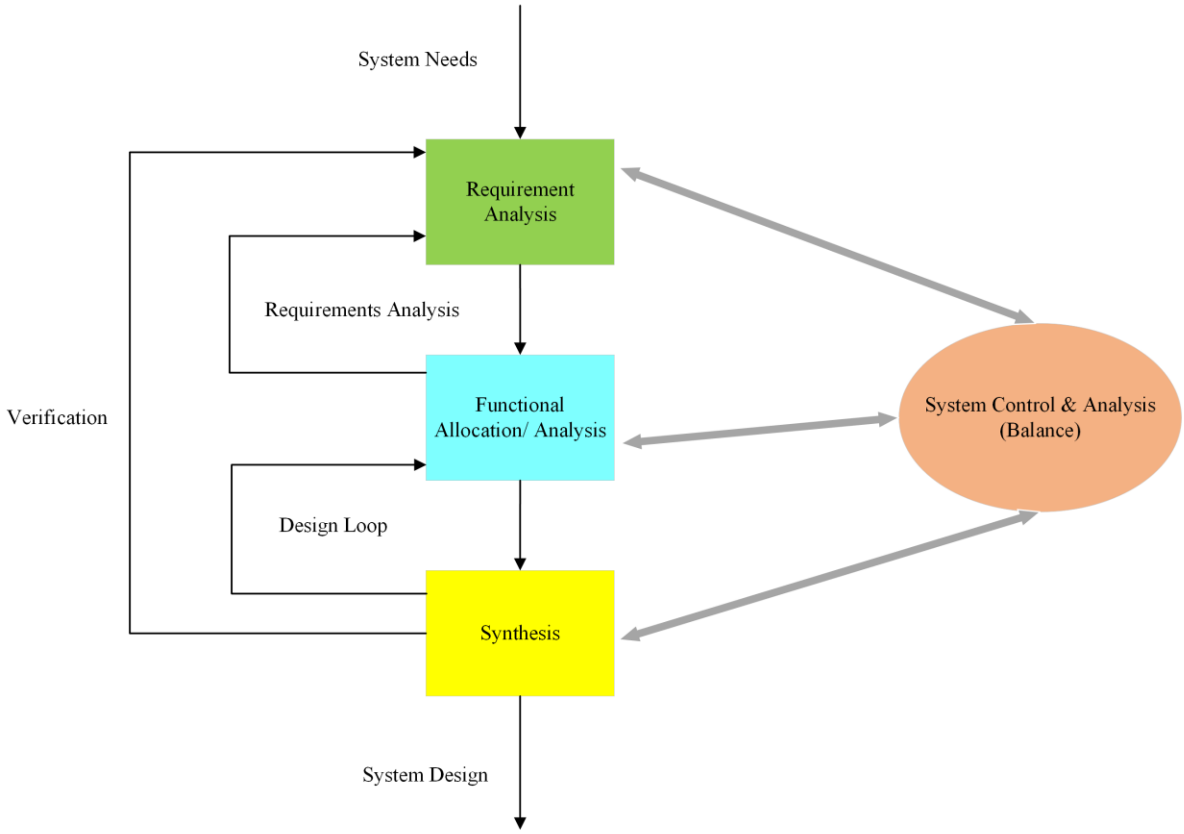

Figure-01: Traditional Systems Engineering Process

2.3 Need For MBSE

Almost all the needs for taking up this methodology comes in the domain of Systems Engineering. Since these engineers need to create and model huge number of requirements and make architectures so that they can be utilized further in the below levels. Therefore, there must be a powerful tool to integrate these so that all the things are expressed in a hierarchal manner.

Some of the needs that were focused while adopting this method were, communication improvement among the stake holders, managing the complex systems in a better manner, understanding the requirement of the systems properly, increasing the productivity by making more people aware of the whole system rather than the concerned part, enhancement of design quality, generating usable products, design reusability in different systems [7].

2.3.1 Existing Notable Methodologies in MBSE

These methodologies are built in such a way that they are user friendly, obey the stakeholders need. The chief objective is to deliver an effective model for the end users [8].

2.3.1.1 Embedded Computer System Analysis Modelling (ECSAM)

This modelling method was built by Lavi and Kudish and was developed for the CBS [9]. ECSAM was built on three views and each view is visible as a dynamic process of developing complex systems. The views are as follows, conceptual architecture, operating modes, and system's capabilities. The operation scenarios and use cases can be built using this modelling method. Most importantly this technique can be implemented at any system level; whether it can be top level or the subsystem level. State charts technology is developed by Prof. David Harel in ECSAM methodology.

2.3.1.2 Model Based Architecture and Systems Engineering (MBASE)

MBASE is built by Prof. Boehm and Prof. Port [10]. It was developed for those software systems that merge product, property, process, and success models. MBASE methodology contains four phases of development, namely inception, elaboration, construction, and transition phase. LeanMBASE is also an extension of MBASE that is a light-weight software. High priority activities, and load balancing are the major activities that LeanMBASE performs.

2.3.1.3 Harmony System Engineering (HSE)

It is developed by IBM and can be used for the projects that include the agile MBSE process [11]. There is a tool kit for the systems engineers known as Harmony SE profile which SE can use in their work of architecture and specification. A lot of common modelling tasks are automated using Rational Rhapsody API. In recent times PIMs are created by SysML or UML that are then converted to AADL models. Now how to make sure that the changes took place during this conversion? Harmony SE methodology provides the solution for it by keeping track of the conversion changes.

2.3.1.4 Object Oriented System Engineering Method (OOSEM)

OOSEM was developed by INCOSE that binds traditional SE practices with the object-oriented concepts [12]. The system analysis and the design information were deployed originally in UML; however, later by the advent of SysML these were modelled to work with SysML. The reason behind it was that UML was majorly used by software engineers and SysML was designed for system engineering applications. This method can perform a huge number of activities including analysis of requirements, architectural design (logical as well as allocated), verification and validation (V&V) of systems and their values.

2.3.1.5 Rational Unified Process System Engineering (RUP-SE)

RUP SE is a software engineering process that was developed by IBM [13]. It is a UML developer environment that has a plugin within RMC. It introduced the locality concept of the model where the connections are specific for every model that is being built. Moreover, the scalability of project is another benefit that it is providing. Some features of this methodology are, it is a repetitive process, tools can support it, the models are configurable, object-oriented approach is supported by this method, and it focuses initially on the architecture of the project.

2.3.1.6 Vitech

Vitech is developed by vitech corporation as a source of systems engineering work [14]. It provides MBSE solutions to its customers. These solutions can be SysML diagrams creation, requirement definition, management, traceability, verification, and validation, DoDAF. Some of the case studies under which Vitech is used are, model-based service volume engineering development using CORE software, privatizing government operations-a systems approach, state machine model of information processing in an operations centre.

2.3.1.7 Jet Propulsion Laboratory State Analysis (JPL SA)

JPL SA was developed by NASA in JPL for applying it in the space missions [15]. This methodology is a kind of MBE. This initiates with a model of system that is to be controlled with all the sensors that are required for interaction to the system. The basic trait that JPL SA has is its ability to separate control systems and system under control. SA arises specific questions along the way of modelling that provides filter of having relevant answers. JPL SA method explains the process of modelling as well as identifying the states of the systems and the relationship between each state, the effect of each state on the system.

2.3.1.8 Object Process Methodology (OPM)

OPM can be referred to as a conceptual modelling language. It demonstrates a system in two views. One is the text, and the other is the diagram or model view. The key building block that comprises this methodology are objects, processes, and states. These three things are linked to each other. The complexity depends on the level of details that one need to put into OPM. The OPM model displays, at any level of detail required, the structural relationships and procedural connections between the system's constituent parts [16]. The single OPM model can be animated in a way that is both clear and expressive, which makes design-level debugging much easier.

2.3.1.9 LITHE Methodology

LITHE was developed by Prof. Ana Lusa Ramos that takes the object oriented top-down approach [17]. This methodology is a part of agile methodology that possess the characteristics of handling complex, huge systems. It is also an MBSE methodology where systems are continuously built and managed along the way of the project. These can be reused for other projects if other projects need that system or some part of that system. LITHE is a merged approach of OOSEM, RUP SE, and OPM.

2.3.2 MBSE for Modern Systems

New systems are developed day by day, as the technology progresses so does the complexity of the systems. For example, the trucks were formerly made by doing the mechanical design; however, this has now been changed with the evolution of measuring the temperature, weather, and other parameters with the sensors. These trucks are loaded with sensors that measure the whole environment around it. This gives rise to many systems and their subsystems in the trucks. The problem arises with accessing the abilities required in the work. Identifying the needs of the system and properly mapping those needs to the required disciplines (Electrical, Mechanical, Fluid Dynamics) can be done using MBSE.

The model of the system is created by elements that are connected to each other. These modelling elements represent the major aspects of our system. The aspects can be behaviour, structure, requirements, and parameters [18]. The existence of geographically dispersed teams, complexity of systems, interdisciplinarity, personalized system description, standards and their integration can be enabled by MBSE. Simple models are a key so that they easily point towards the benefits to the R&D managers who can then accept the model and initiate work on the model development [19].

The industrial complex systems that can be modelled by MBSE method are, telecommunication circuits, factory automation designs, power optimization processes, automobile manufacturing, airplane designs, hardware/software integration models, defence systems, space systems, and many more [20].

Risks analysis and development efforts needs to be initially set for MBSE feasibility in the system. COSYSMO model focuses in identifying accurately the time and effort involved in SE activities that can be furthered to make MBSE assumptions [21].

2.3.3 Benefits of MBSE in SoC

Any methodology that is developed have some unique features. The use of the methodology is at first in a specific field and with the passage of time, the methodology becomes multidisciplinary. MBSE is also the same, the use of it were initially in the mechanical design and then it led to construction of other precise equipment considering the advantages. Some of the benefits that MBSE provides in SoC are:

- Architecture information for the models is centralized to one location.
- The capability of generating automated reports, simulations at several diagrams, conversion to MATLAB/Simulink models poses a huge benefit over the excel, PowerPoint, and Visio illustrations.
- Traceability of requirements is possible as every requirement is modelled and linked to the specific requirement.
- Model reusability is also a benefit of MBSE that enhances productivity, cut operational costs, and accelerates development.
- Adding requirements into already created architecture is extremely easy.
- Documents can be linked to the architecture diagrams to form a sense of consistency.
- Parameters as well as use cases of the project can be managed in centralized manner.
- MBSE can define the software control and co-operation. It enables hardware/software co-design.

2.3.4 Applications of MBSE

Seeing the benefits of MBSE, it is obvious that several industries are implementing this approach. Some of the applications are:

- It is applied in the development of low powered WSN [22].
- It is used for the designing of navigation systems that are integrated together [23].
- MBSE is involved in Mars Sample Return (MRS) Mars Ascent Vehicle (MAV) campaign [24].
- MBSE is applied in the design of agile security in IoT [25].
- It is used in verification of aircraft RMS [26].
- Few of the industries that are applying the methodology are, Audi, AVL, Daimler, Hella, it' OWL SE, INVIRTES, MecPro2, SFB614, Linz, Kannegiesser [27].

2.4 Wymorian Theory (Mathematical Model For MBSE)

This theory was suggested by Albert Wayne Wymore. The theory gives us a mathematical model of the work on developing the model of the system. Basically, he explains it in the terms of a discrete system model. It is as such that the theory consists of five parts, set of states, set of inputs, set of outputs, next states, and the readout function [28]. The set of states is related to readout function in such a way that the readout function is the output of each state. Mathematically it can be written as:

$$Z = \{SZ, IZ, OZ, NZ, RZ\}$$

Here "Z" denotes the name of the system, "SZ" is showing all the states present in the system, "IZ" is defining all the inputs to the system, "OZ" is used for the set of outputs, "NZ" represents the function for the next state, and lastly "RZ" is the readout function [28].

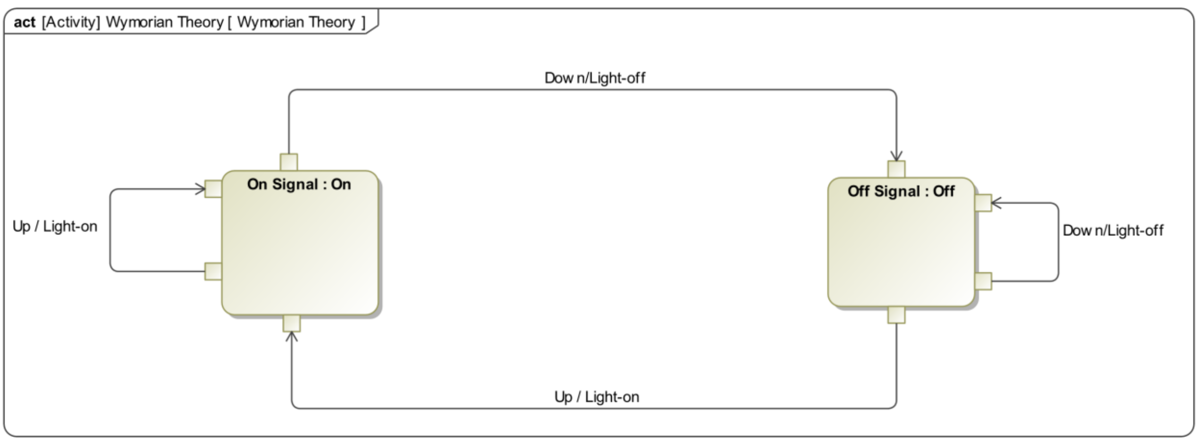

Figure-02: SysML Diagram Capturing Outputs on Transitions

Fig. 02 illustrates the model that is drawn in SysML particularly in Magic Draw tool. It describes the process of a light being on or off. The Wymorian explanation of this figure would be,

Wymorian System (Z) = $\{SZ, IZ, OZ, NZ, RZ\}$	Wymorian two state light Z_2SL= $\{SZ_{2SL}, IZ_{2SL}, OZ, NZ_{2SL}, RZ_{2SL}\}$
States (SZ)	$SZ_{2SL} = \{On, Off\}$
Inputs (IZ)	$IZ_{2SL} = \{Up, Down\}$
Outputs (OZ)	$OZ_{2SL} = \{Light - on, Light - off\}$
Next State Function (NZ)	$NZ_{2SL} = \{((On, Up), On), ((On, Down), Off), ((Off, Up), On), ((Off, Down), Off)\}$
Readout Function (RZ)	$RZ_{2SL} = \{(On, Light - on), (Off, Light - off)\}$

Table-01: Wymorian System Explanation for SysML model

The explanation of every state is given in Table-01. From the table it is evident that simple SysML models can be designed by Wymorian mathematical constructs.

2.5 Languages for Systems Engineering

Every software or tool utilized in the development of ASIC chips requires some language to work in a way as intended. There may be several languages for completing the chip design. The languages can be SysML, UML, SystemC, C++, C#, C sharp and many more.

For systems engineering work there are two languages, namely SysML and UML.

2.5.1 UML

UML is a modelling language used for doing the work on analyzing the system design, behavior, and to keep track of the development in the project. UML diagrams can provide analytics that may help in doing management decisions [29]. UMLx is one tool for doing the system design, planning, and risk management for the project made in UML (there are other tools as well). UML is used in the making of large projects that involves a lot of teams with various resources. UML provides the feature of designing five types of diagrams, Activity Diagrams (act), Package Diagram, Sequence Diagram, State Machine Diagram, Use Case Diagram [30].

2.5.2 SysML

SysML is also a modelling language for system engineer to sort everything of the project in some models. In fact, SysML is an extension of UML as it (SysML) takes the idea of creating classes in it. SysML tools are powerful enough to generate reports from the created models. For this reason, it is extensively utilized in space exploration companies, defense systems, and factory automation industries. The complex processes can be modelled using this language. The detailed description of the use of SysML in a specific process can be seen from this [31] source. This conference paper is applying SysML in MES design. SysML can create nine types of diagrams according to [32], Activity Diagrams, Block Definition Diagrams, Internal Block Diagrams, Package Diagram, Parametric Diagrams, Requirement Diagrams, Sequence Diagrams, State Machine Diagrams, and Use Case Diagrams.

2.6 Modelling Tools for Systems Engineering

System engineering is a broad category to work on and requires several tools to provide best possible needs to its users. Some of the tools that systems engineers work on are listed below:

- Astah
- Enterprise Architect
- MagicDraw
- Microsoft Visio
- Papyrus
- Software Ideas Modeler
- UModel
- Visual paradigm for UML
- StarUML
- ArgoUML

2.6.1 Astah

Astah is created by ChangeVision, Inc. formerly known as JUDE, is not an open-source software application and the programming language used for the development of it is Java. SysML support is partially obtainable in it, automated document generation is a good feature that it supports. Astah can be integrated to Cameo Systems Modeler [33].

2.6.2 Enterprise Architect

Enterprise Architect is created by Sparx Systems and is also not open-source platform for creating diagrams. It is the developed in C++ language. A full support of SysML, OSLC is available in it. This tool can be integrated with Rational DOORS using the MDG link for DOORS [34].

2.6.3 MagicDraw

MagicDraw tool is created by the company named Dassault Systemès. It is not open sourced, and Java is the programming language. SysML, XMI, OSLC support is available. Automated document generation is also possible. It can work on multiple platforms. It can be integrated with Rational DOORS Micro Focus ALM using the Cameo Datahub. Teamcenter integration is also present with other plugins [35]. This integration allows the user to do product lifecycle management (PLM) inside the MagicDraw tool. The PLM operations can be updating, saving, and reusing the project contents.

It is a modelling tool for complex engineering projects that require high level of accuracy for its functions. MBSE methodology can be applied with the help of MagicDraw.

Block based design tools such as Vivado Design Suite is different from MagicDraw. MBSE methodology cannot be applied to Vivado Design Suite, but MagicDraw can do it. Vivado has the functionality of synthesizing and analysing the HDL designs and featuring high level synthesis. This makes the model extremely detailed and complex for the higher management who only wants to see the upper-level flow of the system. MagicDraw provides the systems engineering perspective of the model. Systems engineering perspective is the development and management of the complex system designs. It contains various diagramming windows (BDD, PAR, IBD, ACT) that carries different functionality in the system. Links can be embedded in the diagrams to obtain the source of information.

2.6.4 Microsoft Visio

Microsoft Visio is created by Microsoft, is not an open-source software application. The programming language used for the development of it is not known. SysML support is not available in it. Microsoft Visio cannot be integrated with other tools. It is only a diagramming tool and other functions such as test cases and acquiring results from some equations cannot be modelled in it [36].

2.6.5 Eclipse Papyrus

Eclipse Papyrus is made by the French Alternative Energies and Atomic Energy Commission; it is an open-source platform for creating SysML designs. The programming language is Java, and it supports multiple OS platforms (Windows, Linux, macOS). Automated document generation (excel, word) is a good feature of Papyrus [37].

2.6.6 Software Ideas Modeler

Software Ideas Modeler tool is created by No Dusan Rodina. It is not open sourced, and C# is the programming language on which it is made. SysML, XMI support, and automated document generation are available. It can work on multiple platforms. The licence can be bought for commercial use [38].

2.6.7 UModel

UModel is created by Altova and is also not open-source platform for creating UML, SysML diagrams. It is the developed by the combination of three languages Java, Visual Basic, and C++. A full support of SysML, XMI is available in it. BPMN is also present which makes it easier to model business processes and reaching out to precise decisions for the products [39].

2.6.8 Visual Paradigm for UML

Visual Paradigm for UML is created by Visual Paradigm Int'l Ltd. and is also not open-source platform for creating UML, SysML diagrams. It is the developed by the combination of two languages Java, and C++. A full support of SysML, BPMN is available in it. It is a cross platform environment meaning that it can work in any Java enabled environment [40].

2.6.9 StarUML

StarUML is made by the MKLabs it is an open-source platform for creating UML, SysML designs. The programming language is Java, and it available in Windows, Linux, and MacOS. Automated document generation (excel, word) is a good feature of StarUML. It also provides retina display support to see a particular component in magnified manner [41].

2.6.10 ArgoUML

ArgoUML is released under EPL and the programming language on which it is created is Java. It supports basic Modelling Activities for UML design. Since it is developed in Java, so it features to work on multiple platform that support Java environment. ArgoUML does not support UML 2.0 and needs to upgrade to make its room among its competitors [42].

2.7 Environment for SoC Using SysML

The environment for the development of SoC is summarized as follows [43]:

- DOORS or DOORS Next for defining and managing the requirements
- To model the functional aspects of the system, use MATLAB, UML2
- SysML for the system modelling
- System model execution is done by SystemC

These four points according to the author provide some form of clarity of the use of SysML i.e., MBSE methodology in the development of SoC. In addition to that, since these points presents the use of SysML in system modelling so in accordance with this aspect it is not a new methodology but a distinct implementation of SE method in tools. Moreover, [44] article was published in 2005 and seeing the developments in the telecommunication industry currently I suggest that the adoption of this method will take time; however, the effort put into this will stay for a long time.

In this thesis the author is implementing MBSE approach using the SysML with SoC as the targeted subject of interest. Few articles, journals, and papers are released in this target subject so exploration of SoC in MBSE is needed. Updating and team collaboration in the project at regular intervals is extremely important as changes can affect the IP owners and the block.

SoC is a field where several systems are designed, these systems have many subsystems too. Of course, every system has its own importance, its own requirements, its own features, and its own programmability. These are spread in huge number of tools, organization, and people. This leads to proper organization of documents so that they are not cluttered, easily accessible, and secured. SysML has the ability to do that, with proper support a model of requirements can be created initially and when the project matures, the functionality of every block can be defined.

3 EXAMPLES OF MBSE IN SoC

Since MBSE supports flows, simulations, documents creation, and models conversion so it becomes apparent that it contains several advantages which can be leveraged in the field of SoC design. The examples below reveal some cases of how this approach can be used in radio SoCs.

3.1 Requirements Creation and Management

The customer creates requirements and then architects reform and restructures them for the creation of these complex ICs. This is the first step in every SoC making process. These requirements are written in multiple word documents, explanations are processed in a lot of presentations, and excel spreadsheets are provided which makes the accessibility of these documents in a coherent manner extremely hectic. To eliminate this situation, we can map with models (in SysML), the requirements belonging to different IP owners. In this way we can manage the requirements more efficiently.

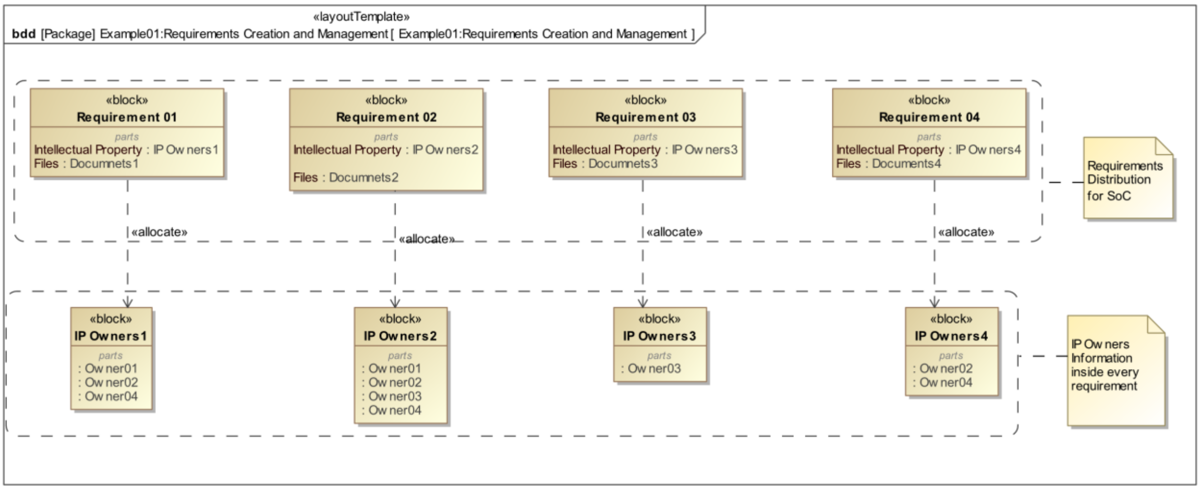

Figure-03: Requirements Creation and Management in MagicDraw

The example use case of MBSE in SoC is represented in the Fig. 03 where there are in total four requirements and each requirement is given a separate block. The IP owners who are linked to the requirement are present below as a subsystem of the requirements. In this way the requirements are properly know to the IP Owners and they can work on that without even looking at the other irrelevant requirements.

For "Requirement 03" in Fig. 03 there is only one IP owner i.e., Owner03; however, for "requirement 01" there are three IP owners i.e., Owner01, Owner02, and Owner04. The requirements are created in BDD, and the IP owners are in the IBD diagram.

3.2 Modelling Specific Links/Connections

SoC is a field where there is a need to specify connections and there must be no ambiguity in the links. These links should be mapped to the correct space so that the misinterpretation can be removed. Mostly, the logical models of the project in MBSE covers this aspect. It explains the proper modelling of the connections in such a way that every system and subsystem have specific links/connections inside them. These can be input, output, as well as internal links/connections between each block. SysML software prompts the user to have those links/connections while plotting.

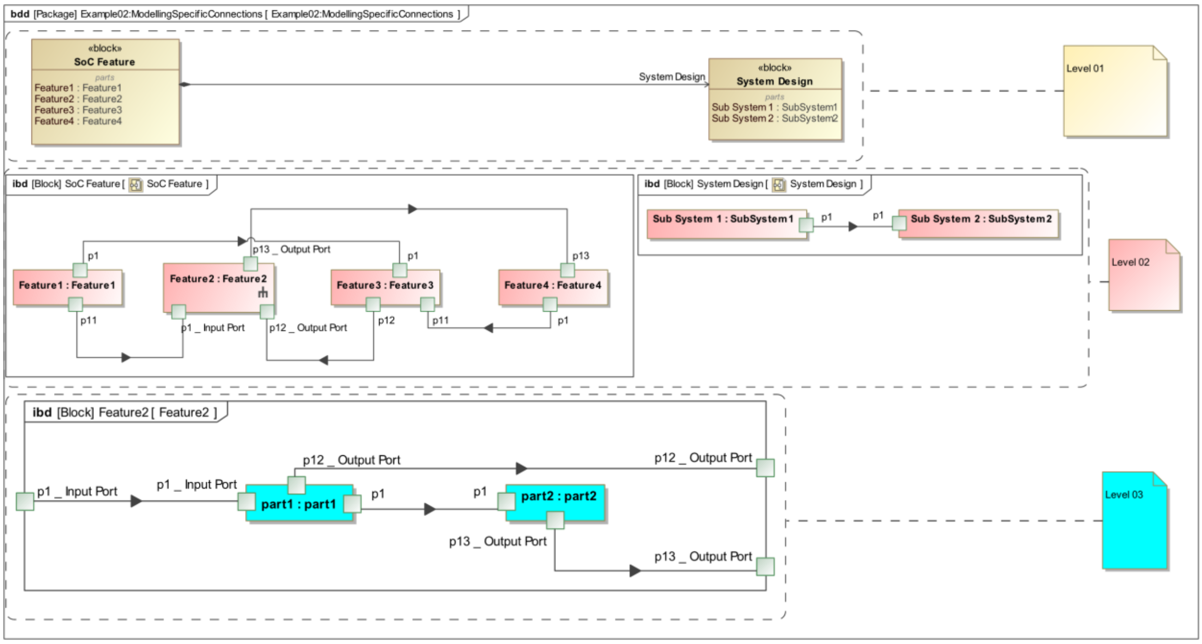

Figure-04: Modelling Specific Links/Connections in MagicDraw

The example for modelling specific links/connections to use while designing SoC is given in the Fig. 04. "BDD" in yellow represents the start of the model or level 01. The "IBD" in pink is the level 02 whose model is made inside the level 01 blocks (SoC Feature, System Design). Level 02 represents proper definition of ports and links. By the properly defined ports and links, it is meant here that in level 02 "Feature1: Feature 1" has "p1" and "p11" as properly defined port as they are connected with a different block, have a unidirectional flow, and contains a specific name in the diagram. The blue "IBD" is the level 03, and it is another block that is made inside the level 02. It also has properly defined ports and links.

If we move from bottom up (level 03 to level 01), we see a relation. In level 03 we have "p1_Input Port" at the left side, "p12_Output Port", and "p13_Output Port" at the right side as input/output pins. These three pins are the same in the level 02 on the "Feature2: Feature 2" block which is making the link/connection between level 02 and level 03, this creates a hierarchical structure of the model where we can even look for which ports to connect while creating links/connections.

Moreover, we see in Fig. 04 that each level has different links/connections within itself, and they are exhibited as a flow that originates from one properly defined port and terminates at another properly defined port.

3.3 Validating Values at Run Time

A tool used in MBSE methodology is MagicDraw which is a SysML tool. MD provides a feature of performing some operations at run time. Validating values is one of those run time features [45]. This can be accomplished by logical expressions. These equations are comparison operators that work to perform the required task and they are written in parametric window of the tool. The constraint block contains the equation or validation condition.

In some cases when doing SoC calculations there comes a need to verify the values. These values can be verified using parametric block in MD tool as appeared in Fig. 05. The following formula is used for the calculation of area:

$$area = 3.14159 \times radius^2$$

Here the "area" value is calculated automatically by the MagicDraw tool when "radius" value is entered. Radius is squared first and then a factor "3.14159" is multiplied to the squared value of radius to get the area value. "3.14159" is the value of "π".

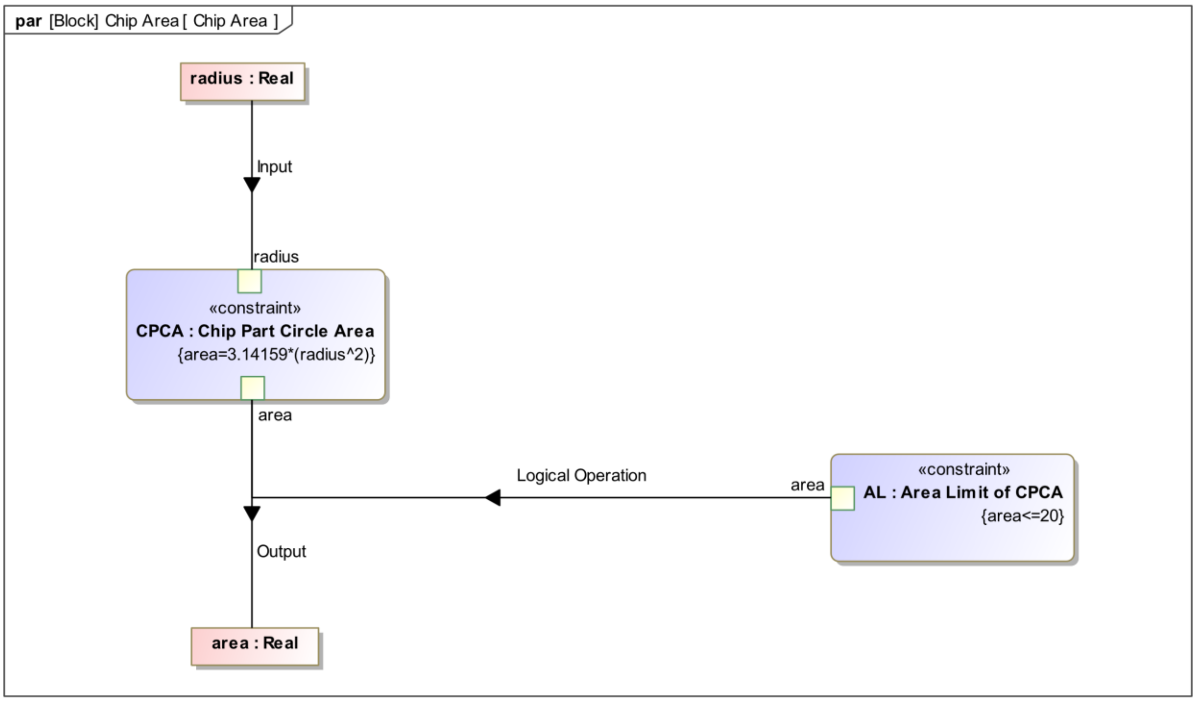

Figure-05: Validating Values at Run Time in MagicDraw

The parametric diagram to represent the logical operation of SysML is illustrated in Fig. 05. Here, the input is taken from "radius" value property and the output is "area" value property. The constraint block named "CPCA" is the block where formula is written. "AL" constraint block contains the condition that CPCA must fulfill to validate the data.

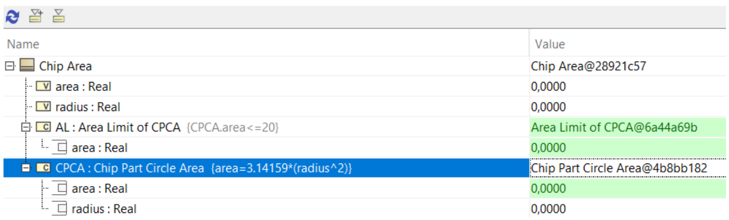

Figure-06: Initiating the Run Command

Initially when the simulation begins the numbers are "0" and the color is green as shown in the Fig. 06. The green color here describes that the digits that are currently written in the radius section are valid.

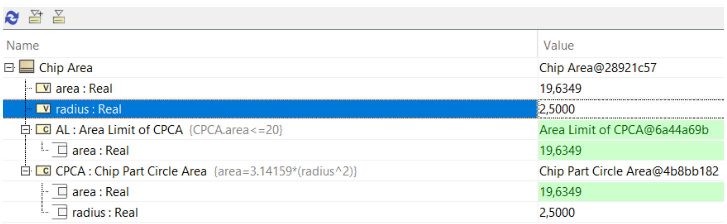

Figure-07: Changing Value of Radius to a Valid Numeral

When the simulation value is changed from “0” to “2.5”, the green color is still intact, it means that the entered number is correct and is visible in the Fig. 07. Moreover, we observe that the value of area is less than “20” and it is satisfying the criteria of our logic.

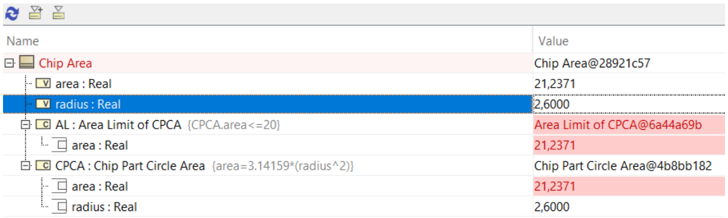

Figure-08: Changing Value of Radius to an Invalid Numeral

When the simulation value is changed from “2.5” to “2.6”, the green color is transformed to red, it means that the entered number here is wrong as seen in the Fig. 08. Additionally, we see that the value of area is more than “20” and it is not satisfying the criteria of our logic. The numbered entered this time is not valid and can be changed and correct number can be stored in an instance table.

This example shows an illustration of validating numerals at run time. We can also say here that if we have more than one component of SoC, and we write the conditions in the parametric diagrams in SysML for those components as such. We can produce the required results that we need.

4 EXAMPLE: ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING

4.1 MBSE Approach

SE can be applied to various hierarchal levels, starting from system level, component level, board level, and finally SoC level [46].

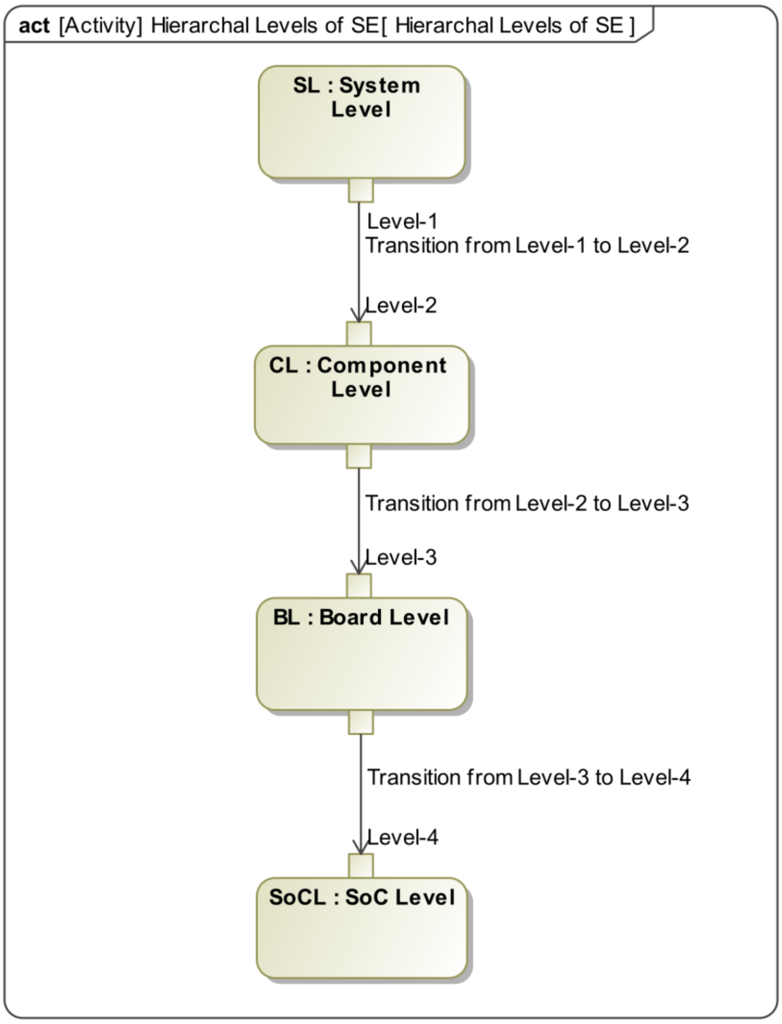

Figure-09: Hierarchical Levels of System Engineering

"System Level" corresponds to the high-level definition of the features and the underlying concepts of these features that are implemented in SoC. "Component Level" is correlated with the use of different types of components in the development of SoC. By the components it is meant here the operating system, timing sources, memory, and voltage regulators. "Board Level" covers the performance as well as functionality mapping of hardware board. Lastly, "SoC Level" refers to the interaction of different intellectual property owners on the common requirements from the customer.

The stages are described in the Fig. 09. The figure displays four phases where SE can be applied. The downward arrow indicates the transition among these levels and the flow is shown here to have an idea that system level is the top level with less details and when we move downwards toward SoC level the details become more and more. These details can be illustrated by the SEs depending on what sort of information is required from SE.

4.2 OFDM

It is a digital transmission method. Rather than applying a single frequency band for transmission OFDM uses multiple closely spaced frequency bands. The information reaches OFDM as a bit stream from upper layers. The signal components of the OFDM signal are known as sub-carriers. Here the sub-carriers are orthogonal to each other and the center of main lobe of each sub-carrier has zero contribution to the other sub-carriers [47]. The data is then encoded and modulated into symbols. QAM or QPSK is used as a modulation method for this purpose [48].

This transmission method provides enhanced efficiency in spectrum, reduced complexity, and higher reliability in the networks [49]. OFDM carries the data in parallel, it carries several closely spaced subcarrier signals, that are orthogonal to each other, side by side [50]. OFDM is used in 4G LTE networks, 5G NR eMBB data transmission, and the improved version of it can be utilized for low latency networks in beyond 5G and 6G.

OFDM is available in the downlink path of physical layer, the input is the time domain data (TDD), and the output is phase adjusted time domain data (PA TDD). This modulation transforms the data that is coming as a time domain data into frequency domain data. The frequency data is then phase shifted to make several subcarriers so that maximum data can be travelled in the given bandwidth. The phase shifted version of frequency modulated data is then reconverted into time domain data. This data is then sent for the transmission from one antenna carrier. Along the way of OFDM, phase compensation is additionally added to match the Tx and Rx frequencies.

4.3 OFDM Top to Bottom Approach

Representing top to bottom approach in SoC is the main challenge which SysML is eliminating with the addition of accessing documents at a central location in a modelled manner. MBSE needs some form of requirements to be useful for system engineers initially. Whenever, a new project requires some components of previous project that is made in SysML, MBSE make sures that those components are present at the correct location.

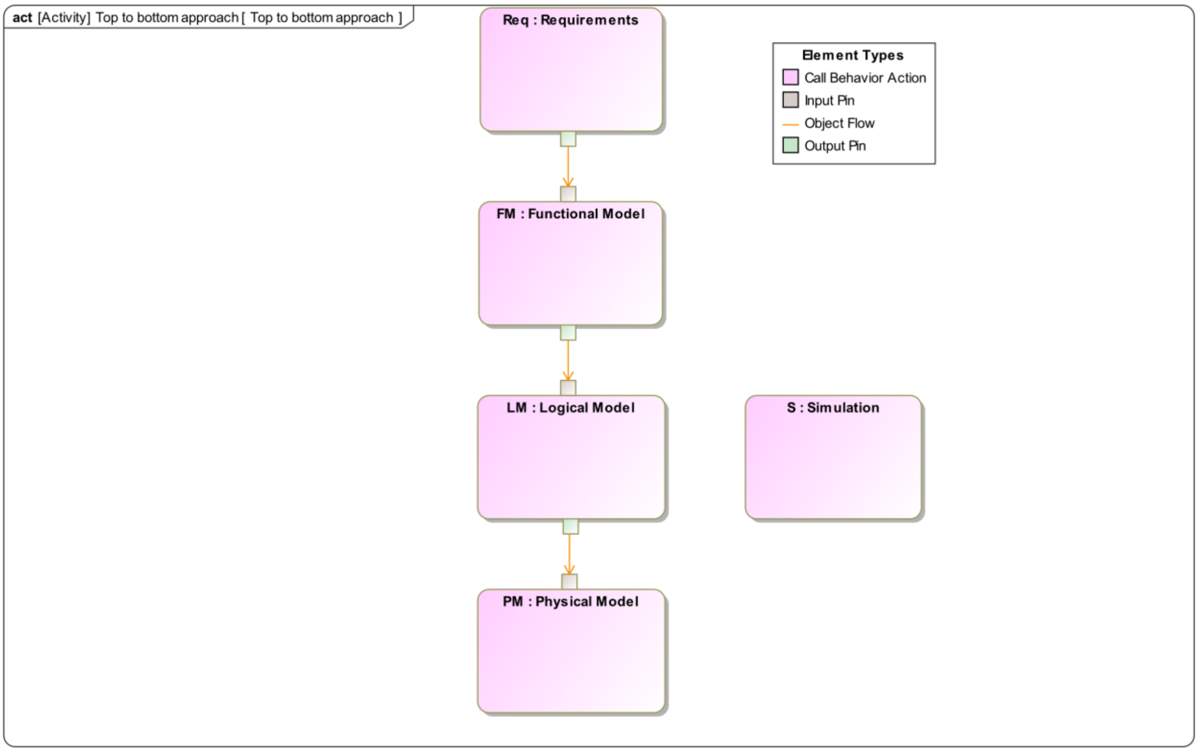

Figure-10: Overview of Top to Bottom Approach

Presenting top to bottom approach here means that requirements are already made under which the project is going to be built which is the top level of the hierarchy. The next one is the creation of functional models to illustrate the upper-level flow of the project as seen in Fig. 10. After this the logical model is given where a detailed interface and flow of blocks is described that is updated as the project matures. Physical model reveals the parametric equations and values that are present in the requirements, additional parameters can be made in the specific block for more details.

Since the logical model provides deep description of the subsystems so multiple simulations are made in this level; therefore, a different block can be made as a side-by-side activity to it as expressed in Fig. 10.

4.3.1 Requirements

Requirements are made by the customer initially. The requirements can be unclear to the technical experts. To define it properly, industries have specialist in this area who make these requirements which are well defined with proper flow to the sub-systems. The OFDM specific requirements can be made in the Word file, requirement data can be stored in sheets, and presentations can be made for proper illustration of the workflow initially. The requirements for OFDM are:

- IFFT in the data stream path
- Pre-scaling and post-scaling component
- CP addition and bit reversal with phase compensation

4.3.2 Functional Models

In these models the features present in the requirements are exhibited. OFDM specific functional model contains pre-scaling block, IFFT block, post-scaling block, bit-reversal block, cyclic prefix block, and phase compensation block. There can be other blocks with sub sub-systems. Therefore, these models portray the functionality of the system or subsystems depending on the use. Fig. 11 and Fig. 12 depicts the functional model of the OFDM symbol generation.

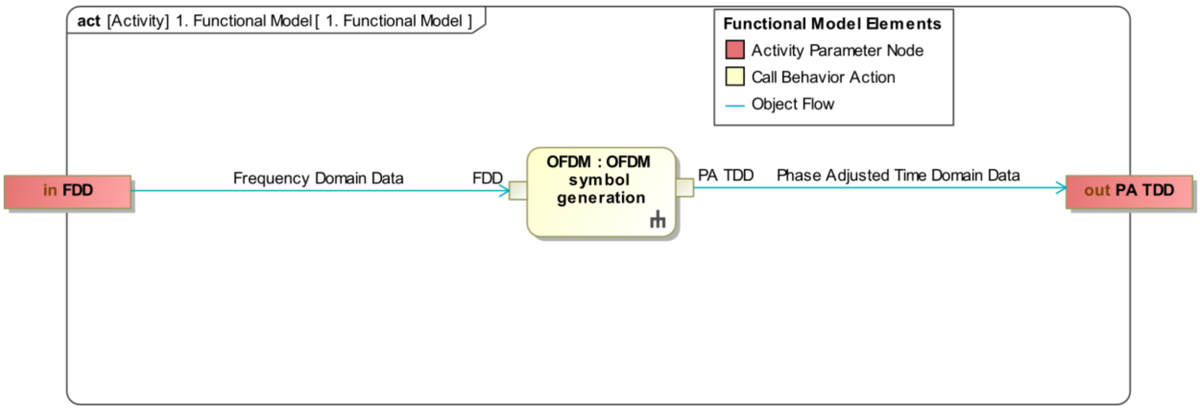

Figure-11: OFDM Symbol Generation block in Functional Model

Fig. 11 expresses the initiation of the process where we have an OFDM symbol generation block in activity diagram that is the system of interest in our case. The input (in red color) is taken from FDD that is the frequency domain data coming from a different block. The output (in red color) is PA TDD which describes the phase adjusted time domain data. We are generating OFDM symbols in the OFDM symbol generation block.

The legend titled as Functional Model Elements in Fig. 11 indicates the elements existing in the model. There are in total four elements. Since the input/output interfaces are descriptive so these are not displayed in the legend. The red color blocks are the activity parameter nodes, the yellow is the call behavior action, and the cyan color interprets the object flow or data flow.

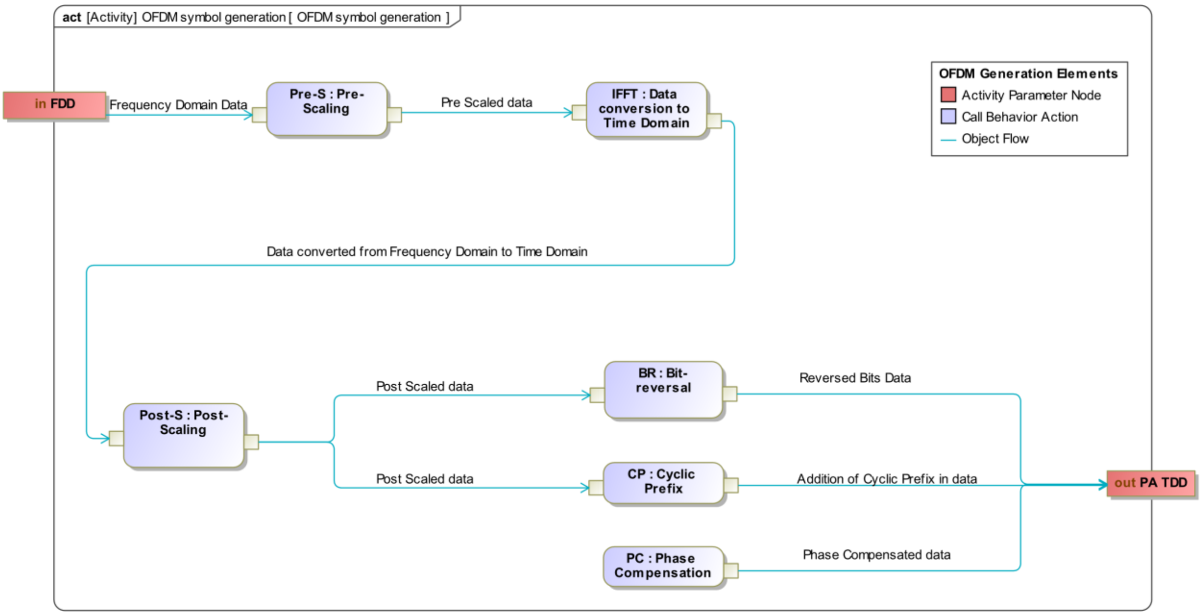

Figure-12: Detailed View of OFDM Symbol Generation as a Functional Model

The work that the OFDM generation block will do is presented in Fig. 12. How it will do the work will be observed in the logical model later. The model in Fig. 12 is also an activity diagram and the input and output are the same as in Fig. 11. The FDD data is fed into pre scalar before conversion that scales the high frequency signal to low frequency signal. The data is then passed through IFFT for conversion from frequency domain to time domain. Then it is post scaled. After that CP is added to remove ICI. The bits are reversed, and phase compensation is added to make the Tx and Rx frequencies the same. After going through all these steps, the output data will be phase adjusted time domain data and is sent to the other blocks for further processing.

The legend termed as OFDM Generation Elements in Fig. 12 exhibits the elements of the model. As the input/output interfaces are just internal connections so these are not shown in the legend. Apart from it there are three elements or blocks present in the activity diagram. The red color blocks are the activity parameter nodes, the purple ones are the call behavior actions, and the cyan color represents the object flow or data flow.

4.3.3 Logical Models

In these models the process along with data streams are presented. It contains the software development part of the SoC. OFDM specific logical model contains pre-scaling block, serial to parallel converter block, IFFT block, parallel to serial converter block, post-scaling block, bit-reversal, cyclic prefix block, and control stream. There can be other blocks with sub sub-systems. Hence, the following models explains the logical view of our system of interest. Fig. 13, and Fig. 14 expresses the logical model of the OFDM symbol generation.

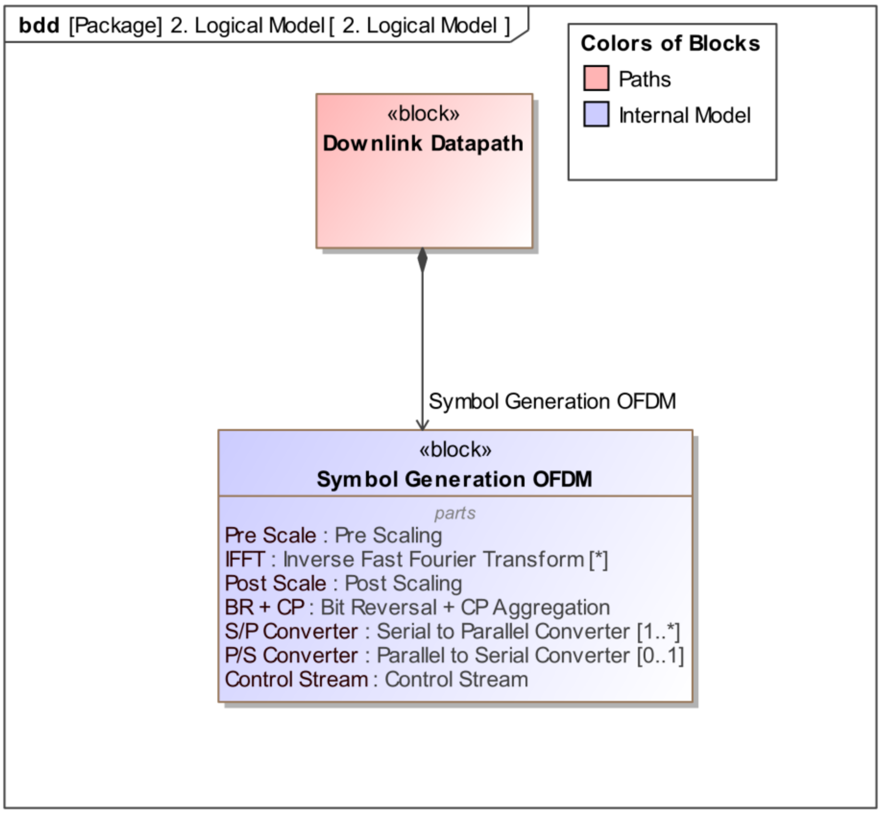

Figure-13: Logical Model Overview of OFDM Symbol Generation in MagicDraw

Fig. 13 indicates the logical model's overview where we have a subsystem with the name Symbol Generation OFDM, and Downlink Datapath block in block definition diagram. Generation of OFDM symbol comes under the category of Downlink Datapath which is the reason for connecting both these blocks in Fig. 13. The bottom left corner of the Symbol Generation OFDM can contain an icon that will represent that there is an IBD existing inside it.

The legend named Colors of Blocks in Fig. 13 illustrates the necessary elements used in the above model. The red color block is the path where our subsystem of interest is available, and the purple color interprets the internal model Symbol Generation OFDM block.

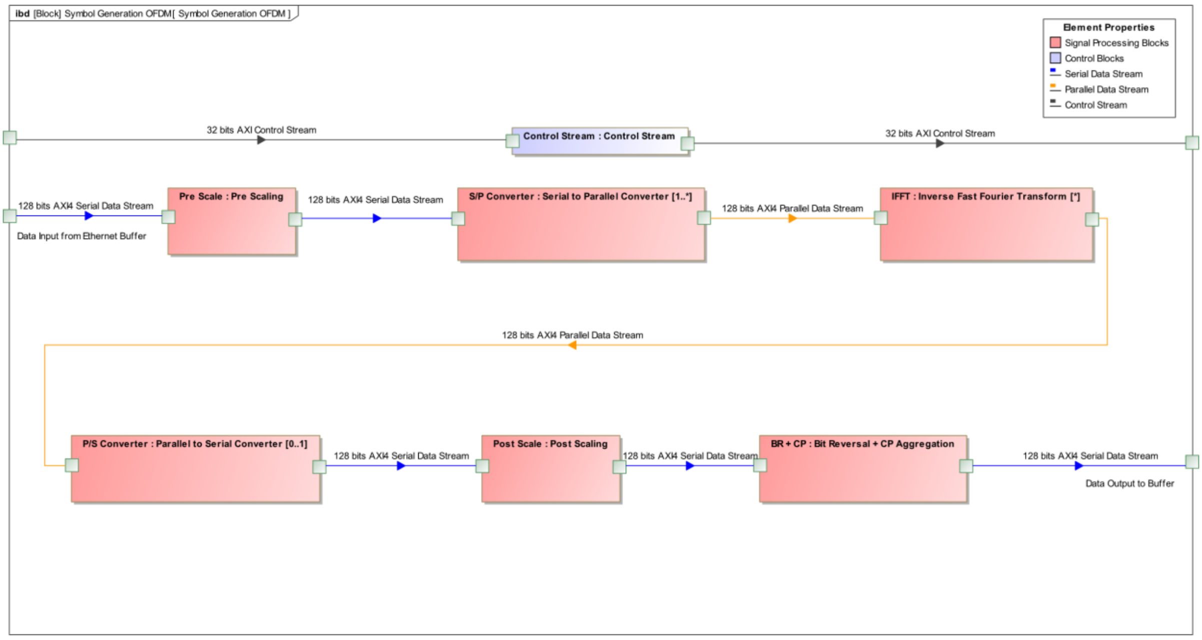

Figure-14: Detailed View of Logical Model Inside Symbol Generation OFDM Block in MagicDraw

The logical model of OFDM symbol generation looks like Fig. 14. The proxy ports are properly mapped that indicates that the model has end to end connectivity. The data stream changes here from serial to parallel in one block, parallel to serial in another block, and various operations are performed in it. It means that the signal processing blocks shown in red color contains some activity for the data stream. 32 bits AXI data is sent to the control stream block that is presenting that apart from 128 bits data there is a total of 160 bits of data which will flow through the Symbol Generation OFDM part.

The symbols are input in a serial manner and pre scaling is done to the data stream, a serial to parallel converter converts this serial data into parallel. Serial data is a high data rate stream and when it is converted into parallel, it is transformed into low bit streams. Then the data is sent to compute the Fourier transform as shown in Fig. 14. Specifically Inverse Fast Fourier Transform (IFFT). "K" data symbols are sent correctly to IFFT to obtain the right output. This process refers to the frequency domain processing of the data. Time domain processing comes after taking the output from IFFT. In time domain processing the parallel data is transformed to serial data. Then post scaling is done and cyclic prefix (CP) is added to have the protection from ISI. Finally, the data is sent to the channel for further processing.

More IBDs' can be created inside this Fig. 14 model, since the work of architecture and specification is to identify the key details hence additional elements can be added by other teams that are building the SoC.

The legend labeled as Element Properties in Fig. 14 reveals the elements existing in this model. The orange color lines are the parallel data streams, the blue color lines contain the serial data stream path, the black color lines illustrate the control stream path, the purple block is the control stream block, and the red color represents signal processing blocks.

S/P converter block has [1..*] that shows this block has one to many connections, IFFT block has [*] representing many to many connections, and lastly P/S converter block has [0..1] illustrating many to one connection.

Fig. 13, and Fig. 14 are interlinked in such a way that Fig. 14 can be accessed directly from Fig. 13 by going inside the Symbol generation OFDM block. Additionally, the parts available in Symbol Generation OFDM that are pre scale, IFFT, post scale, BR+CP, S/P converter, P/S converter, and control stream in Fig. 13 are the blocks used in Fig. 14.

4.3.4 Physical Models

As the logical and functional models possess some unique abilities to sketch the structure of the models, physical models also provide a different set of unique features for representing the models. In these models the parameters of the systems and subsystems are illustrated. A challenge arises for outlining the parameters in such a way that the MBSE approach focuses on top to bottom hierarchy. While building the top architecture, the parameters are not so many, and they need to be present to make the project end to end. Physical model removes this challenge by incorporating parameters and simulated values. These models contain the hardware part of the SoC.

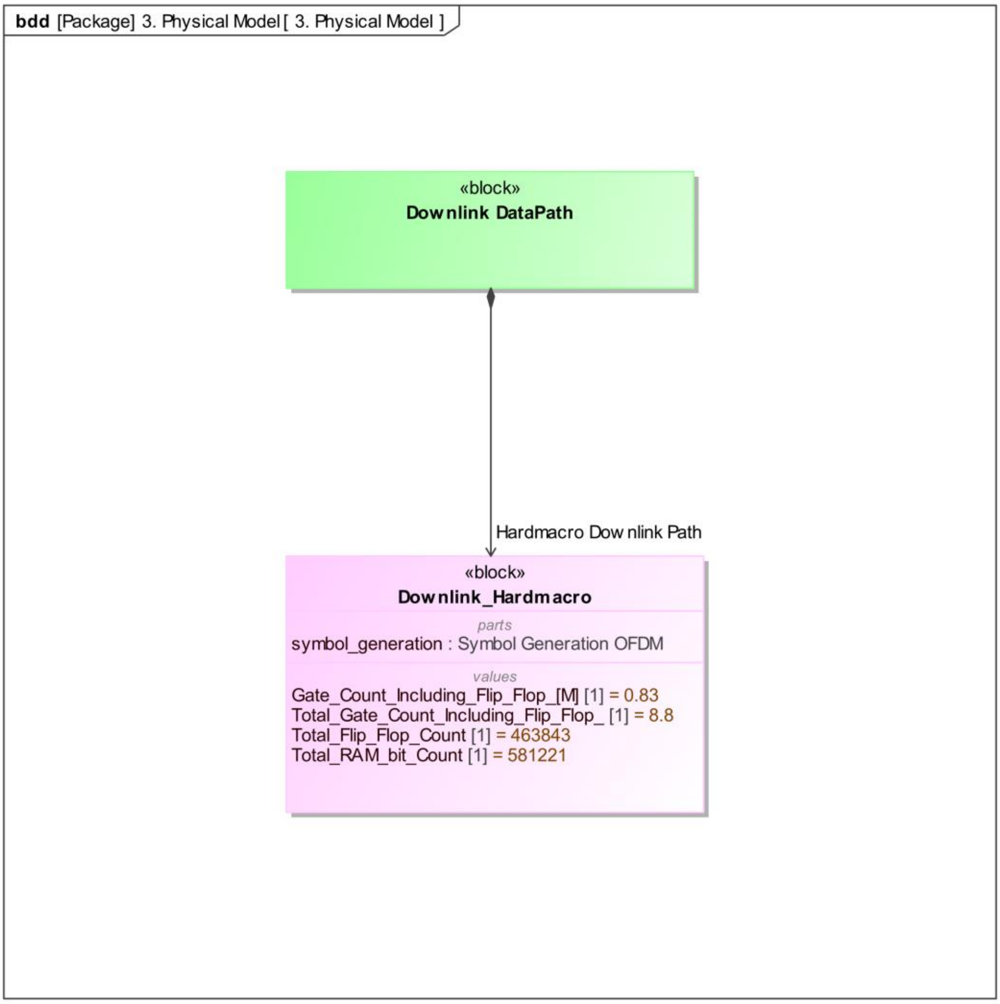

Figure-15: OFDM Symbol Generation Physical Model Block Definition Diagram in MagicDraw

The physical model of OFDM symbol generation is exhibited as a BDD in the Fig. 15. It is more like parametric values, but it is termed as physical model as it contains parts information as well. These numeric numbers can be verified by comparing them with the place where they are taken from. A parametric diagram can also be made inside the "Downlink_Hardmacro" block as expressed in Fig. 05 in MagicDraw to validate the parameters. For this a known quantity needs to exist when we run the physical model so that whenever a value changes, it can be observed.

Fig. 15 represents top to bottom approach in such a way that there is a top block represented in green and that block has some "symbol_generation" part shown in pink which contains different values. The pink block illustrates that "Downlink_Hardmacro" block contains some parametric values.

4.4 MATLAB Integration into MagicDraw

MATLAB simulations can be integrated into this model as MagicDraw is compatible to do simulation in MATLAB. For doing simulations on the signal data or obtaining data for visual-ization, MATLAB is a handy tool. The available signal data can be taken from MagicDraw and put into the MATLAB simulation and vice vera. These tools can be run side by side. The benefit of it is that we can view the whole model along with the part where we want the simulation and see the effect of it on the entire model. Parametrize the blocks data correctly, feed it into the model and observe whether the data obtained is correct or not. We also notice that whenever a parameter is updated to a new numeral, it is changed everywhere it is used in the model and not just at that place. Below example will present how it is done.

4.4.1 Example of MATLAB Integration

The example illustrates MATLAB integration into MagicDraw. For this purpose, a block of physical layer of OSI model is taken in consideration. This block is the OFDM symbol gener-ation block where the code in MATLAB is generating a set of numbers and figures for the realization of signals and noise in a channel where it is receiving random number of bits. The code file is accessed via MagicDraw, and some parameters are created to obtain the required data in the code.

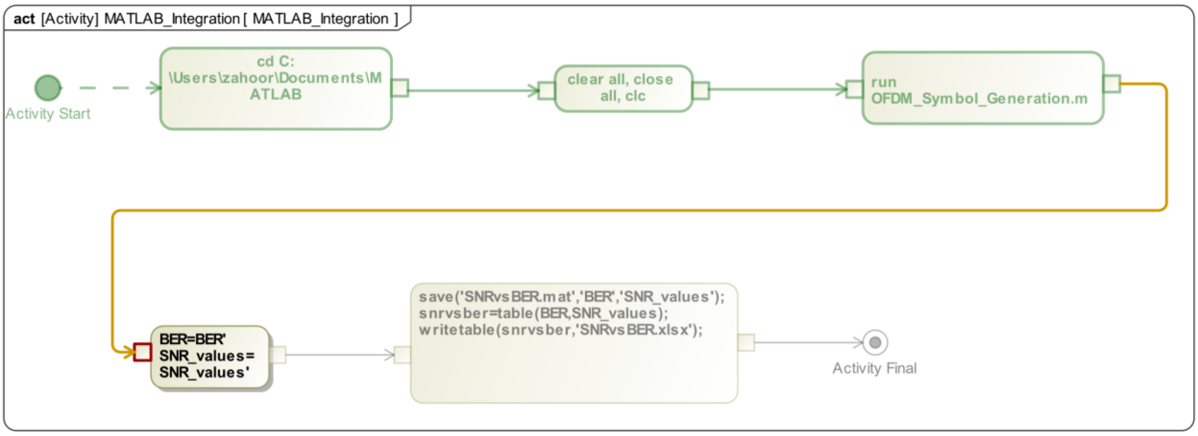

Figure-16: Activity Diagram for MATLAB Integration in MagicDraw

Fig. 16 demonstrates the activity diagram to acquire values from MATLAB using MagicDraw. All the blocks appeared here have the property of simulating in a MATLAB en-vironment (which is done in the backend). The activity initiates with the “Activity Start” ele-ment and accesses the location of the file in the second block/element. Then the MATLAB script for clearing the data is indicated in the third block. Fourth block is the main block in this activity as it contains the exact file name and the run command for running the script. The next block is extracting and outputting the desired values (which in this case are BER and SNR_values), transposing them and then storing in a variable. Moving further in the simulation is the block which is saving the file in two extensions. One extension of file is a .mat file that stores the data for use by MATLAB and the other is a .xlsx file and takes data into an excel sheet. Then the activity is finalized and end at “Activity Final” element.

The green lines and blocks in Fig. 16 express that the activity has been completed. The yellow lines explains that the data is present at this stage and the red lines show where the simulation is proceeding. The blurred blocks and elements predict that the activity has not yet arrived at this stage.

4.4.1.1 Simulation Results

As the simulation in Fig. 16 steps into different blocks, any type of error or execution in the activity arriving in the block is referred to the console window of MagicDraw. The simulation is performed in such a manner that there seems to be no errors arriving in the activity. Activity is initialized, started, and terminated successfully with the required data.

BER Formula: 20 log_{10}	SNR Unit: dB
-0,145984775	-30
-0,324468239	-28
-0,324468239	-26
-0,294465136	-24
-0,324468239	-22
-0,506696384	-20
-0,415104432	-18
-1,343762628	-16
-1,974467075	-14
-3,435397705	-12
-6,435704345	-10
-9,201758308	-8
-10,45757491	-6
-10,55462496	-4
-10,55462496	-2
-10,55462496	0
-10,55462496	2
-10,55462496	4
-12,76544328	6

Table-02: SNR (dB), BER (20log_{10}) Values Observed in MagicDraw

Once the simulation in Fig. 16 is complete, the data in Table-02 is produced. BER and SNR data is the required data that we need here. This data can be seen in the console window of MagicDraw. The data is stored in an excel sheet where it is updated every time the simulation is performed. Therefore, we can obtain the updated data from the corresponding excel file as the file is linked to the simulation. If there comes a need to process the old data, new file can be made to have the updated data and the previous file can be disconnected.

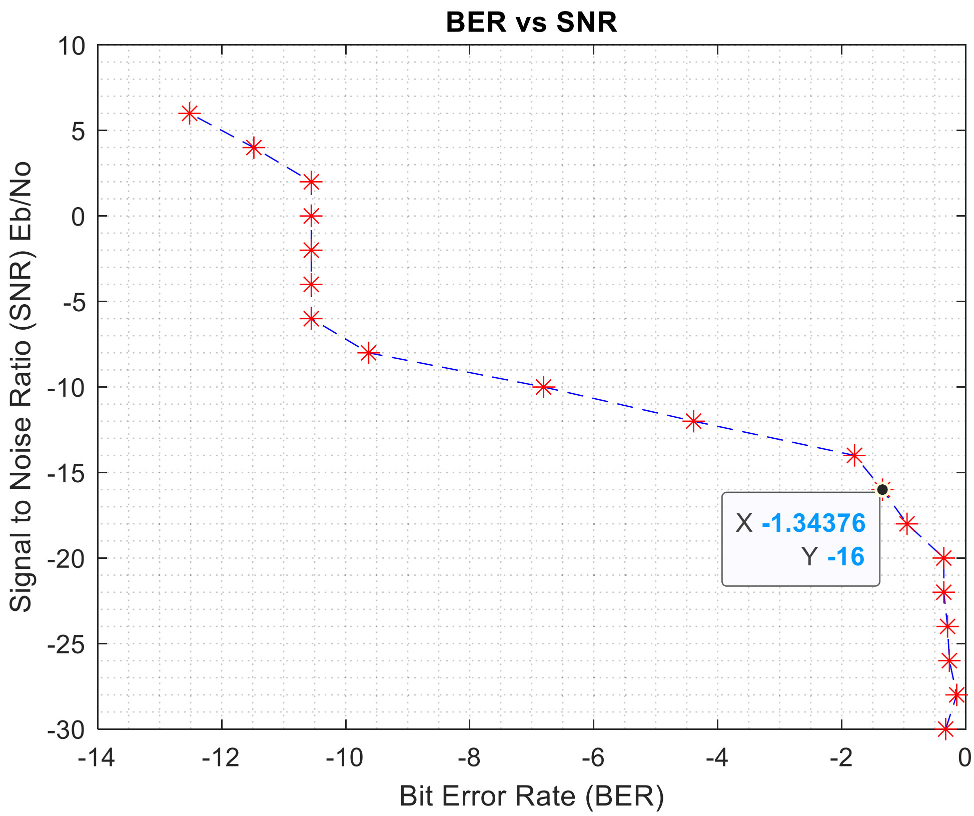

Figure-17: BER vs SNR plot Obtained from MATLAB Code Simulation

The activity diagram simulation also generates a plot of "BER vs SNR" illustrated in Fig. 17. It takes the MATLAB code, runs it, and reveals the plots in a MATLAB window. The x-axis exhibits BER numbers ranging from -14 to 0 and y-axis presents SNR numbers from -30 to 10. The code is made in such a way that it is taking SNR data at every two steps and BER data at random intervals. The total data points for the plot are nineteen. Red points indicate the values that are appearing in table-02 and the blue dotted lines express the binding of the points and can be referred to as intermediate values in those points. A data tip is also displayed to reveal that when SNR value is "-16" BER is at "-1.34376".

To use the data of the Table-02 in MagicDraw we need an instance table. The instance table is created in MagicDraw. This table can be built by syncing the excel file and mapping the columns of excel sheet to the desired values of the instance table. Such as BER data in instance table is mapped to BER data in the excel sheet and similar for SNR data.

Instance table is similar to excel sheet; however, validations on these numbers can be performed by making it a value property in MagicDraw. It means that once this table is created the rows are made as instance specification and are visible in the containment tree of the project. As the instance table created from Table-02 has nineteen instances hence there will be this number of instance specifications.

4.4.1.2 Validation of Simulated Results

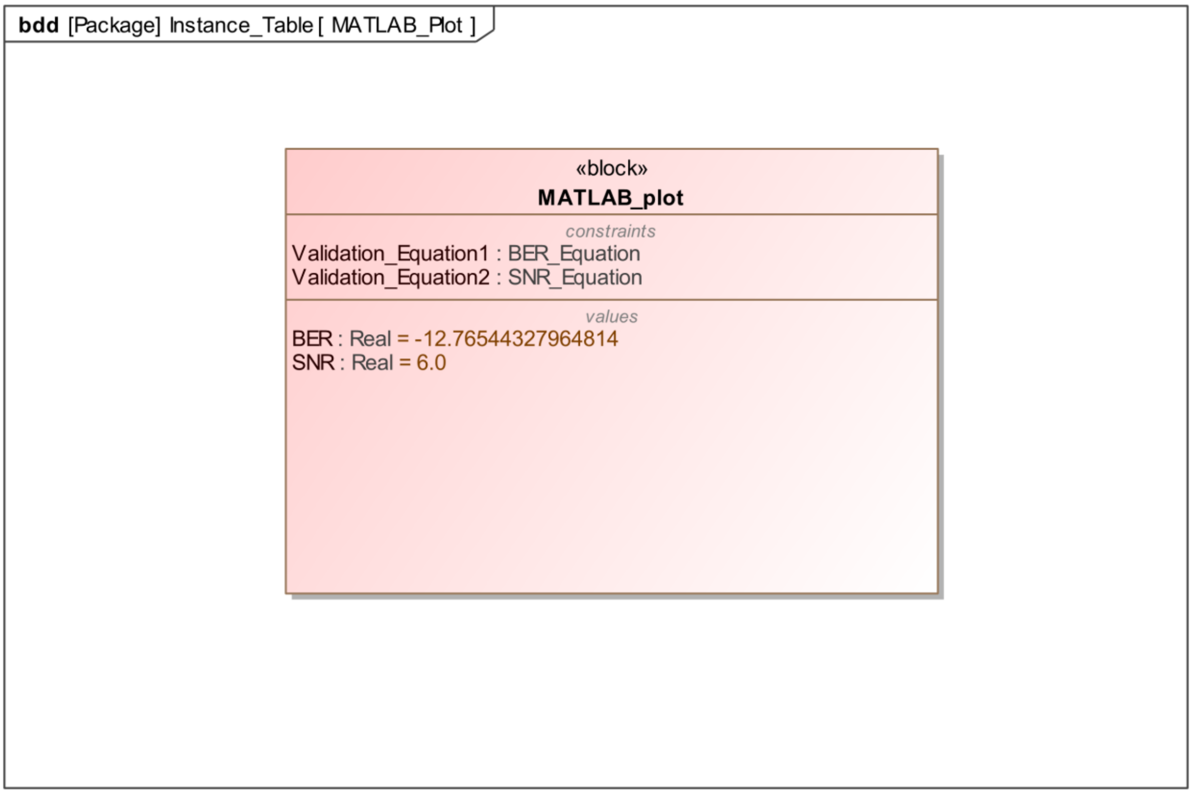

Figure-18: Block Definition Diagram for Validating Values in MagicDraw

The instance specifications that are made from table-02 can be used for doing comparisons of BER and SNR values. These values can be verified or validated using a parametric diagram inside a block. As expressed from Fig. 18, there are two equations below the constraint heading (referred to as constraint blocks) for validating BER and SNR data with the names "Validation_Equation1: BER_Equation" and "Validation_Equation2: SNR_Equation". Fig. 18 also takes two parameters "BER" and "SNR" (referred to as value property) with the default values as indicated in the corresponding figure.

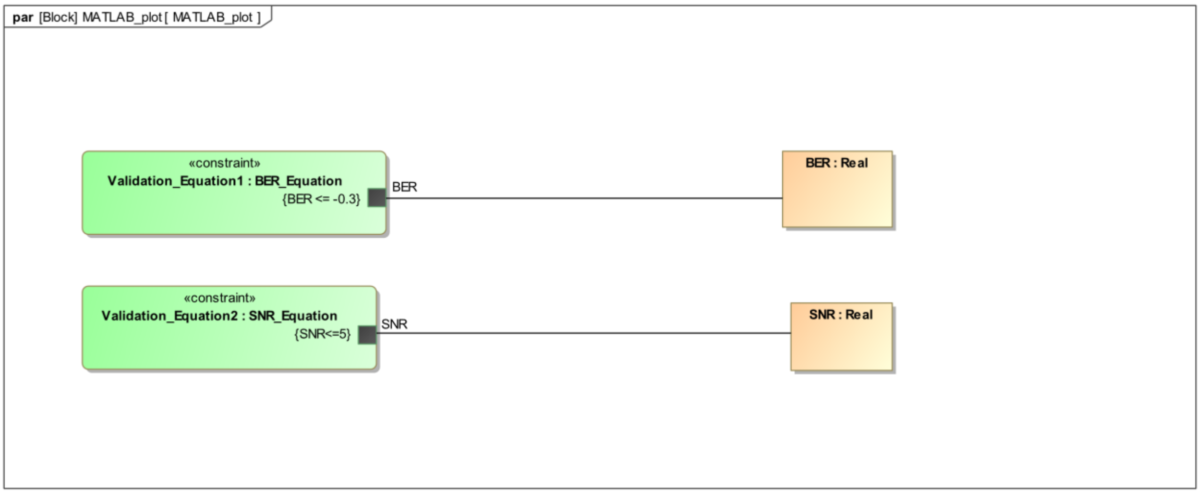

Figure-19: Parametric Diagram Containing Validation Equations and Slot Values in MagicDraw

Fig. 19 illustrates a parametric diagram that is made inside the BDD in the Fig. 18. To validate values over here, it is meant that when the BER or SNR value reaches above a certain threshold a warning is generated with the red colour. Additionally, the parametric equations (or constraint property) of Fig. 19 are,

$$BER \leq -0.3$$

$$SNR \leq 5$$

It means that whenever a BER value greater than -0.3 or SNR value more than 5 is inserted, an error is generated and can be observed with the red colour. A true value is seen in green.

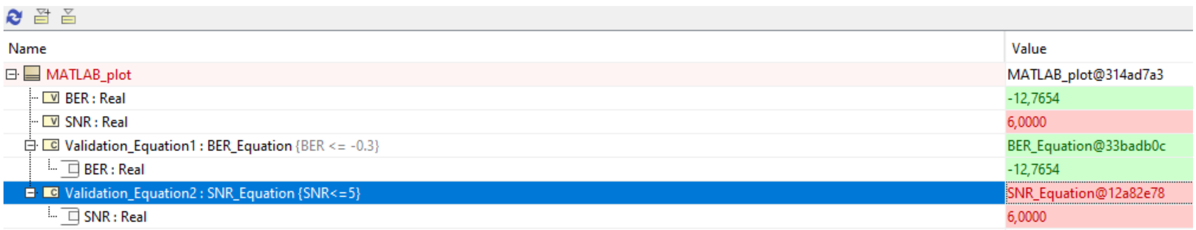

Figure-20: Outcome of the Parametric Diagram Simulation in MagicDraw

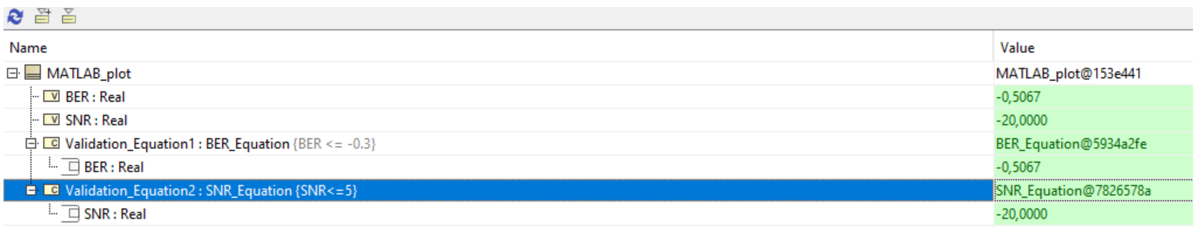

Figure-21: Outcome of the Parametric Diagram Simulation in MagicDraw

The effect of the condition of parametric equation can be observed in the Variables tab in MagicDraw when the simulation runs. Fig. 20 and Fig. 21 is presenting the results of the two simulations performed with two different input values of BER and SNR. Commonalities between the two simulations are that the name column have the same set of variables, value properties, and equations. Differences are in the value column of this tab where there are different values of BER, SNR, and different identification (ids') of the constraint blocks.

In the first simulation (shown in Fig. 20) BER value is in range of the threshold; therefore, a green highlighted section is displayed. In the similar way a red highlighted section for SNR reveals that the input number is exceeding the threshold and is not a valid one. Conclusion to this observation is that the SNR value needs to be different than 6.

In the second simulation (shown in Fig. 21), BER and SNR numerals are in the range of threshold; hence, a green highlighted value is appearing for both. We deduce that these values are valid and can be used further in the simulations.

Similarly, we can produce these sorts of results for the other seventeen instances and describe their behaviour. Valid data can be stored and used whenever there is a need. Use cases can be built on top of these and projects can be tested on those use cases.

5 DISCUSSION

MBSE has paved way to bridge the gap between validating simulations and returning to the architecture. It is as such, one can do simulations alongside architecture in different software. Since lots of software are used in the making of SoC so MBSE can provide a way to link the software and make everything connected end to end. The possibility of linking everything of a project related to architecture in a single containment tree is producing immense help to the companies. Now validation engineers can verify easily whether the designer has created the design in accordance with the architecture as they have a single up to date architecture available. Test cases development and their effect on the overall architecture can be monitored in a simple way using MBSE. MBSE methodology development tool creators need to figure out a way to keep the flexibility in such a way that every organization can use it in their work while keeping in view the MBSE principles. Additionally, in this way it can get more market of the tool that is used in the process.

Every block of SysML has different set of components that help in building scenarios of the architecture. For example, parametric diagram can contain constraint block, value property, and simulation configuration can be linked to the diagram for obtaining values in the value properties. In fact, every block, every file, every instance can be connected, and hyperlink can be created that makes the sense of everything linked to each other. This is one of the properties of MBSE.

Optimization of architecture can be done using MBSE in an easy way. First there should be a need to optimize and then location of points where optimization can be done. Once it is established then a set of procedures can be performed on the SysML model for optimizing the diagrams. In addition to that, if those optimizations or part of the optimization is not needed and the architects prefer previous version then it can be made accessible for everyone.

The advantages are explained in the thesis and there is a limitation to working with MagicDraw. One constraint block can only contain one equation, correspondingly the equation cannot take a set of values or a vector to display validation errors on all those numbers. Every value must be set as default before validating it in the parametric diagram.

6 SUMMARY

Practical solutions of utilizing MBSE are available in many industries and every industry has their own ways of expressing systems in SysML language. This research was focused on applying MBSE in radios SoC. Initially, traditional systems engineering aspect was explored and limitations were found in it, then the need for MBSE was demonstrated in detail that is posing the solution to traditional SE. The use of MBSE methodology in modern systems was explained along with benefits and applications. A theory related to MBSE was discussed as well. Moreover, languages plus some tools were presented that paved way for utilizing MagicDraw in the thesis. Some examples for using MBSE in SoC are also exhibited. SysML was the language used and the tool is MagicDraw for demonstrating the architecture of the system. MagicDraw supports several external evaluators and MATLAB is one of them which is producing simulation results in this thesis and feeding into the MagicDraw tool. A component that is OFDM symbol generation had been taken in consideration for the purpose of interpretation and creation. This thesis has described how a SoC process can be implemented in one complete model with top to bottom approach containing functional, logical, and physical model of the OFDM symbol generation along with validation of simulation results.

7 REFERENCES

[1] Pastor, Ó. (2007, April 11). Model Transformation Technologies in the Context of Modelling Software Systems. Retrieved September 12, 2022, from https://drops.dagstuhl.de/opus/volltexte/2007/865/

[2] *SysML Open Source Project - What is SysML? Who created it?* (n.d.). SysML.org. Retrieved September 13, 2022, from https://sysml.org/

[3] *SysML FAQ: What is the current version of SysML?* (n.d.). SysML.org. Retrieved September 13, 2022, from https://sysml.org/sysml-faq/what-is-current-version-of-sysml.html

[4] *SysML FAQ: How should SysML be applied to MBSE project?* (n.d.). SysML.org. Retrieved September 13, 2022, from https://sysml.org/sysml-faq/sysml-applied-mbse.html

[5] Rogers, E., & Mitchell, S. (2021). MBSE delivers significant return on investment in evolutionary development of complex SoS. *Systems Engineering*. 24: https://doi.org/10.1002/sys.21592

[6] Mizine, I., & Wintersteen, B. (2022). Multi-level hierarchical system approach in computerized ship design. 1-2.

[7] Rao, H. B. (2013, August 16). *Study of Modern Modelling Techniques for Model Based Systems Engineering Methodologies*. IJERT.

[8] Rao, B. H., Padmaja, K., & Gurulingam, P. (2013). A brief view of Model Based Systems Engineering Methodologies. J. Eng. Trends Technology (IJETT), 4(8), 3266-3271.

[9] Lavi, J. Z., & Kudish, J. (2004). "Systems modeling & requirements specification using ECSAM: a method for embedded computer-based systems analysis," Proceedings. 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, 2004, pp. 2-11, doi: 10.1109/ECBS.2004.1316676.

[10] Koolmanojwong, S., & Boehm, B. (2007). "An Empirical Study on MBASE and LeanMBASE," First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), pp. 496-496, doi: 10.1109/ESEM.2007.53.

[11] Zhang, T., Wu, L., Qi, & Xu, H. (2012). "Architecture analysis and design language & harmony system engineering process," 2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC), pp. 7D2-1-7D2-12, doi: 10.1109/DASC.2012.6382421.

[12] *Object-Oriented SE method*. (n.d.). INCOSE. Retrieved September 27, 2022, from https://www.incose.org/incose-member-resources/working-groups/transformational/object-oriented-se-method

[13] Kruchten, P. (2002). "Tutorial: introduction to the Rational Unified Process®," Proceedings of the 24th International Conference on Software Engineering. ICSE 2002, pp. 703.

[14] *Case Studies – Vitech*. (n.d.). Retrieved September 27, 2022, from https://www.vitechcorp.com/case-studies/

[15] Mission Data System/State Analysis Public Web Site, Jet Propulsion Laboratory, California Institute of Technology, (June 2010).

[16] Bolshchikov, S., Renick, A., Mazor, S., Somekh, J., & Dori, D. (2011). "OPM Model-Driven Animated Simulation with Computational Interface to MATLAB," 2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 193-198, doi: 10.1109/WETICE.2011.32.

[17] Ramos, A. L., Ferreira, J. V., & Barceló, J. (May 2013). "LITHE: An Agile Methodology for Human-Centric Model-Based Systems Engineering," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 43, no. 3, pp. 504-521, doi: 10.1109/TSMCA.2012.2207888.

[18] Friedenthal, S., Moore, A., & Steiner, R. (2008). A Practical Guide to SysML, The Systems Modeling Language. Burlington: Morgan Kaufmann/OMG, Elsevier.

[19] Little, J. (1970). “Models and managers: The concept of a decision calculus,” Manag. Sci., vol.16, no. 8, pp. B466–B485.

[20] Davidz, H., & Turner, R. (2022, May 20). *Systems Engineering Implementation Examples*. SEBoK. https://www.sebokwiki.org/wiki/Systems_Engineering_Implementation_Examples

[21] Valerdi, R. (2008). The Constructive Systems Engineering Cost Model (COSYSMO): Quantifying the Costs of Systems Engineering Effort in Complex Systems, Germany, Saarbrcken:VDM Verlag.

[22] Taha, J., Salehi, V., & Abraham, F. (2018). "Development of a Low Powered Wireless IoT Sensor Network based on MBSE," 2018 IEEE International Systems Engineering Symposium (ISSE). pp. 1-8, doi: 10.1109/SysEng.2018.8544420.

[23] Yang, L., Cheng, N., & Li, Q. (2018). "The Design of Integrated Navigation System Based on MBSE," 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC). pp. 1-5, doi: 10.1109/GNCC42960.2018.9018855.

[24] Rountree, I. (2022). "MBSE Applications for the MSR SRC Mars Ascent Vehicle," 2022 IEEE Aerospace Conference (AERO). pp. 1-14, doi: 10.1109/AERO53065.2022.9843502.

[25] Papke, B. L. (2017). "Enabling design of agile security in the IOT with MBSE," 2017 12th System of Systems Engineering Conference (SoSE). pp. 1-6, doi: 10.1109/SYSOSE.2017.7994938.

[26] Ju, C., Wang, S., & Wang, T. (2020). "Research on verification technology of aircraft RMS based on MBSE," CSAA/IET International Conference on Aircraft Utility Systems (AUS 2020). pp. 171-175, doi: 10.1049/icp.2021.0260.

[27] Tschirner, C., Dumitrescu, R., Bansmann, M., & Gausemeier, J. (2015). "Tailoring Model-Based Systems Engineering concepts for industrial application," 2015 Annual IEEE Systems Conference (SysCon) Proceedings. pp. 69-76, doi: 10.1109/SYSCON.2015.7116731.

[28] Wach, P., & Salado, A. (2019). Can Wymore's mathematical framework underpin SysML? An initial investigation of state machines. Procedia Computer Science, 153, 242–249. https://doi.org/10.1016/j.procs.2019.05.076

[29] Qi, K., & Boehm, B. (2018). "Poster: UMLx: A UML Diagram Analytic Tool for Software Management Decisions," 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion). pp. 278-279.

[30] Wikipedia contributors. (2022, August 24). *Unified modeling language*. Wikipedia. https://en.wikipedia.org/wiki/Unified_Modeling_Language

[31] Pietrac, L. (2011, September 5). *On the use of SysML for manufacturing execution system design*. Archive ouverte HAL. https://hal.archives-ouvertes.fr/hal-00767991

[32] Wikipedia contributors. (2022a, April 29). *Systems modeling language*. Wikipedia. https://en.wikipedia.org/wiki/Systems_Modeling_Language

[33] Astah. (2022, July 27). *Premier diagramming, modeling software & tools*. https://astah.net/

[34] Enterprise Architecture. (2020, June 9). *Enterprise Architect Diagrams Gallery*. https://sparxsystems.com/resources/gallery/diagrams/architecture/arc-main-page.html

[35] *MagicDraw - CATIA - dassault systèmes®*. (2022, July 1). MagicDraw. https://www.3ds.com/products-services/catia/products/no-magic/magicdraw/

[36] Microsoft. (2016). *Flowchart maker and diagramming software | microsoft visio*. Microsoft Visio. https://www.microsoft.com/en-us/microsoft-365/visio/flowchart-software

[37] Papyrus. (2020, June). Eclipse Papyrus. SysML, UML design tool. https://www.eclipse.org/papyrus/

[38] Rodina, D. (2021, July 27). *Diagram CASE tool for software modelling & analysis - UML, BPMN, ERD*. Software Ideas Modeler. https://www.softwareideas.net/

[39] Altova GmbH. (2020, March 17). *UModel UML modeling tool - altova.* https://www.altova.com/umodel

[40] Visual paradigm for UML. (2020, July 23). *Visual Paradigm*. https://www.visual-paradigm.com/solution/freeumltool/

[41] *StarUML*. (2022, June 14). StarUML. https://staruml.io/

[42] *ArgoUML*. (2011, December 15). ArgoUML. https://argouml-tigris-org.github.io/tigris/argouml/

[43] Müller, R., Eisenmann, H., & Ried, O. (2005, August). Systems Engineering methodology applied on a distributed system-on-chip development. In *DASIA 2005-Data Systems in Aerospace* (Vol. 602), pp. 4.

[44] Müller, R., Eisenmann, H., & Ried, O. (2005, August). Systems Engineering methodology applied on a distributed system-on-chip development. In *DASIA 2005-Data Systems in Aerospace* (Vol. 602).

[45] *Logical expression*. (n.d.). No Magic Product Documentation. Retrieved September 23, 2022, from https://docs.nomagic.com/display/CST190/Logical+expression

[46] Müller, R., Eisenmann, H., & Ried, O. (2005, August). Systems Engineering methodology applied on a distributed system-on-chip development. In *DASIA 2005-Data Systems in Aerospace* (Vol. 602).

[47] Mahmud, M. H., Hossain, M. M., Khan, A. A., Ahmed, S., Mahmud M. A., & Islam, M. H. (2020). "Performance Analysis of OFDM, W-OFDM and F-OFDM Under Rayleigh Fading Channel for 5G Wireless Communication," 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS). pp. 1172-1177, doi: 10.1109/ICISS49785.2020.9316134.

[48] *What Is OFDM?* (n.d.). MATLAB & Simulink. Retrieved October 13, 2022, from https://se.mathworks.com/discovery/ofdm.html

[49] Sun, D., Zheng, B., & Xu, X. (2010). "Multi-user cognitive OFDM with adaptive subcarrier and power allocation," 2010 2nd International Conference on Future Computer and Communication. pp. V1-159-V1-163, doi: 10.1109/ICFCC.2010.5497815.

[50] Mahmud, M. T., & Ryu, H. G. (2021). "Performance Evaluation of OFDM Hybrid Number and Index Modulation for 6G Mobile System," 2021 International Conference on Information and Communication Technology Convergence (ICTC). pp. 39-42, doi: 10.1109/ICTC52510.2021.9621057.