International Journal of Computer Applications (IJCA)
Volume 1, Issue 1, Jan–June 2019, pp.12–17, Article ID: IJCA_01_01_002
Available online at
https://www.iaeme.com/ijca/issues.asp?JType=IJCA&VType=1&IType=1

MODEL BASED SOFTWARE PROCESS

Romita Biswas
Safety Engineer, Lyft Level 5, Palo Alto, CA, USA

Palak Talwar
Senior Safety Engineer, Lyft Level 5, Palo Alto, CA, USA

ABSTRACT

This paper aims to provide an understanding of what the Model Based Software Process is and why it is critical to embed it early in the software development process. The Model Based Software Process is best used in highly complex safety-critical applications with short development cycles. It lays the framework to allow for dynamic growth and traceability from the development to validation phase. The motivation behind this process is to reduce the risk of software failures to as low as reasonably practicable

Key words: Software Process, High Level Process, Test Case

Cite this Article: Romita Biswas and Palak Talwar, Model Based Software Process, *International Journal of Computer Engineering and Technology* 1(1), 2019, pp. 47-52.
https://www.iaeme.com/ijca/issues.asp?JType=IJCA&VType=1&IType=1

1. INTRODUCTION

Objective

This paper aims to provide an understanding of what the Model Based Software Process is and why it is critical to embed it early in the software development process.

Purpose

The Model Based Software Process is best used in highly complex safety-critical applications with short development cycles. It lays the framework to allow for dynamic growth and traceability from the development to validation phase. The motivation behind this process is to reduce the risk of software failures to as low as reasonably practicable.

Advantages

- Enable the early and efficient detection of faults/failures in work products (ISO - 26262 Section 6)
- Continuity of information shared across software life cycles and different teams
- Easy traceability of requirements to implementation
- Allows for Rapid Iteration

- Resource-efficient time critical development
- Represents more than one work product: requirements, architectural design, model verification, code verification (ISO - 26262 Section 6)
- More efficient test case generation (ISO - 26262 Section 6)
- Highly automated testing (ISO - 26262 Section 6)
- Potential for formal verification techniques (ISO - 26262 Section 6)
- Integrates automotive industry standards for model and code verification
    - ISO-26262
    - MISRA C
    - MAAB (MathWorks Automotive Advisory Board)
    - AUTOSAR

2. HIGH LEVEL PROCESS

The Model Based Software Process includes constructing and analyzing requirements, developing Simulink model iteratively, integrating model verification in model based development process, building test cases, and performing all levels of testing. It reflects the close relationship between development and testing. The process attempts to integrate automotive industry standards for the creation of high integrity software.

The traditional V-Model demonstrates the relationship between each phase of the development cycle and its corresponding phase of testing. V-Model is the most traditional model followed for software development and management.

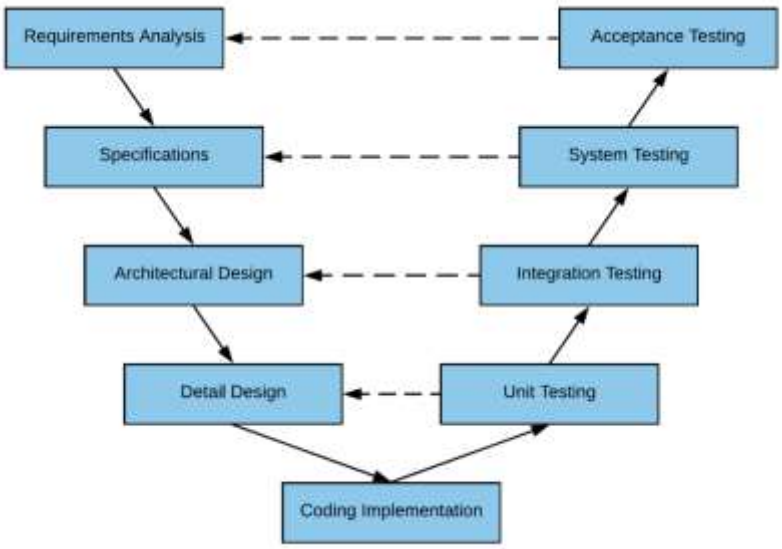

Figure 1 Traditional V-Model Process

Model Based Software Process follows a similar flow where requirements are gathered and analyzed, a high and low level design is created, and testing is performed. The requirements phase includes both requirements analysis and specification. Requirements are traced directly from the requirements management tool into Simulink. Once imported in Simulink, those requirements can be directly traced to where they are implemented in the model using Simulink Tools. Once that model has been verified using Simulink Checks and Unit Testing, the code is generated. That automated code undergoes code verification again to ensure compliance. The rest of testing is performed after that.

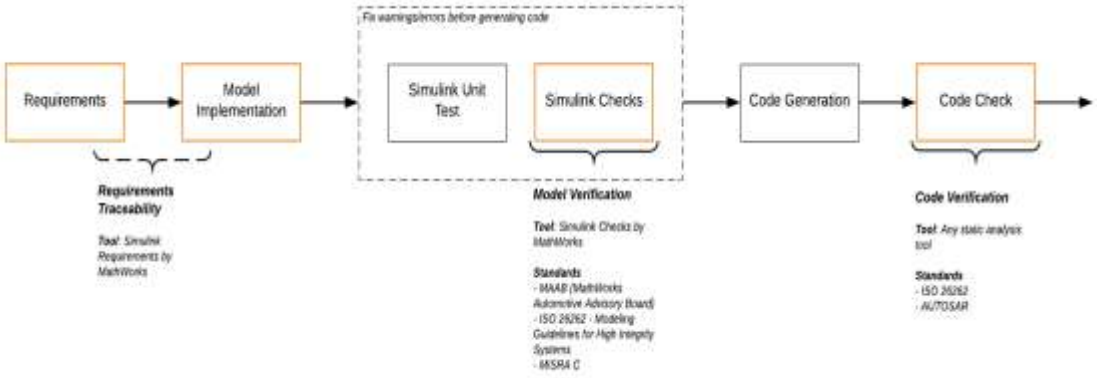

Figure 2 Model Based SW Process (Left Hand Side of V)

In practice the Model Based Software Process follows a more integrated workflow in order to achieve a high integrity system. This is where the Model Based SW Process becomes the Advanced Spiral V-Model, which is customized to allow for rapid iteration while being constrained by time and manpower. Three components of the process provide structure and stability: Requirements Traceability, Integrated Model Verification, and Final Software Verification. Simulink Tools and Code Verification Tools aid in creating software that is compliant and high integrity.

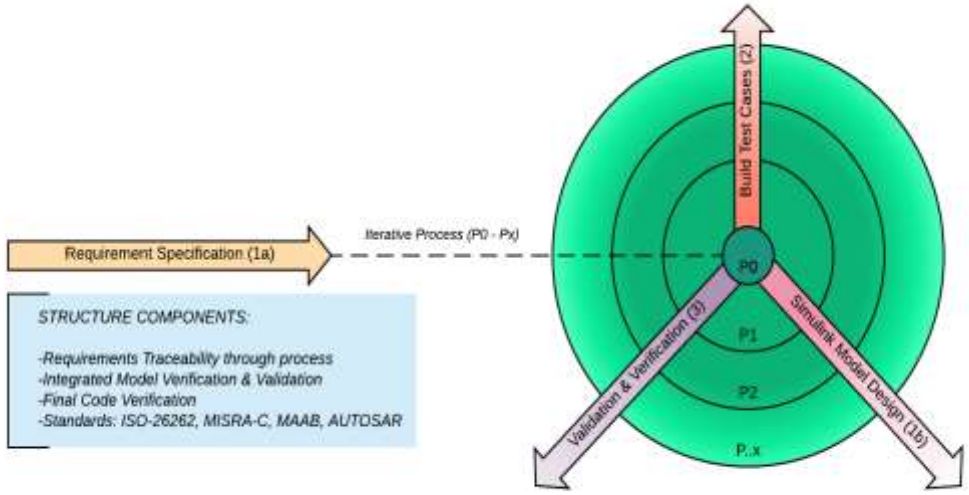

Figure 3 Advanced Spiral V-Model

An overview to the workflow, and the overall function of each branch is discussed here. Each branch is discussed more in depth in The 3 Branches Section.

In the Requirements Specification stage, requirements are created and verified by key stakeholders. The process starts with the P0 tasks and breaks down into 3 branches that support each other to create models iteratively, ex: P0 to P1 to Px.

Entering Branch 1, the Simulink Model Design is implemented starting with a high level design and then each component is developed to create a low level design. Each Simulink Requirement is linked to a component in the model to provide Requirement Traceability.

During implementation, Simulink Checks provides compliance checks so model verification is integrated into the design phase.

Simultaneously entering Branch 2, test cases are built early in the process. This provides the advantage of creating stronger requirements. Test cases are directly created from requirements to test the model. Once the desired implementation has been completed, the model undergoes Simulink Unit Testing. Then Code is generated, and the rest of testing is performed (Branch 3).

The 3 Branches

- Branch 1a: Requirement Specification
- Branch 1b: Simulink Model Design
- Branch 2: Build Test Cases
- Branch 3: Validation & Verification

3. REQUIREMENT SPECIFICATION (1A)

The Requirements Management Process includes drafting an initial set of requirements, having requirements reviewed by relevant stakeholders in the requirements management tool, holding formal peer reviews of requirements, and baselining requirements after they are considered technically sound, reasonable, and verifiable. Defining requirements involves coordinating with several groups.

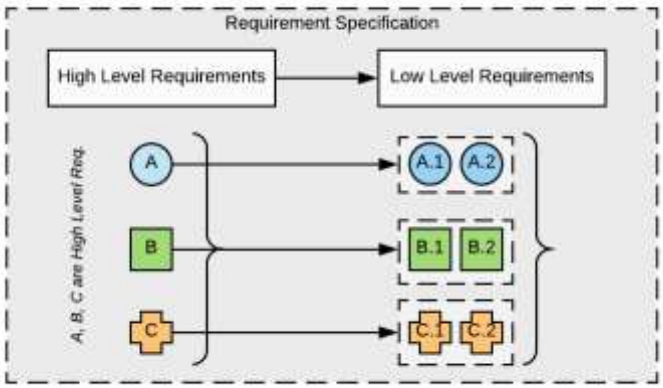

Figure 4 Requirement Specification

In the Requirement Specification stage, High Level and Low Level Requirements are created and both are verified respectively by key stakeholders.

Simulink Model Design (1b)

Once the high-level requirements have been reviewed completely by key stakeholders, the high level architectural design can be developed. Each component designed in the high level design can be linked to high level requirements using Simulink Tools. The components or blocks can be developed individually addressing a set of lower level requirements. Each block is linked to a requirement in both directions. For example, changing a requirement will highlight blocks that are linked to it, and changing a block will highlight the requirement it is linked to. So there is complete requirement traceability through the model design without needing to document links and changes externally.

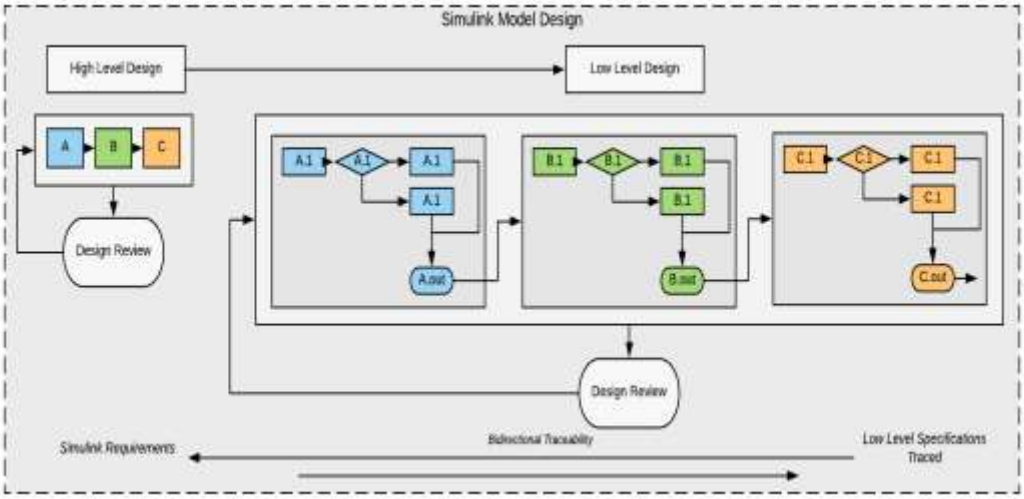

Figure 5 High Level Design to Development of an Individual Block

As individual blocks are created, the Simulink Checks perform compliance checks in real time for each block, minimizing costly rework and additional testing late into the process. Simulink Checks can be configured using Model Advisor to check if the model is compliant with modeling standards.

4. BUILD TEST CASES (2)

Building test cases early in the development process rather than after model implementation, provides the advantage of detecting poor requirements and design failures early on. Test cases are directly created from each level of requirements. Test cases can be further refined and modified as the design and implementation of the model proceeds, thus creating robust testing and stronger requirements. Building test cases in parallel, also allows analysis of test coverage. Simulink requirements can be linked to each specific test case and distinguish which blocks and test steps apply to it.

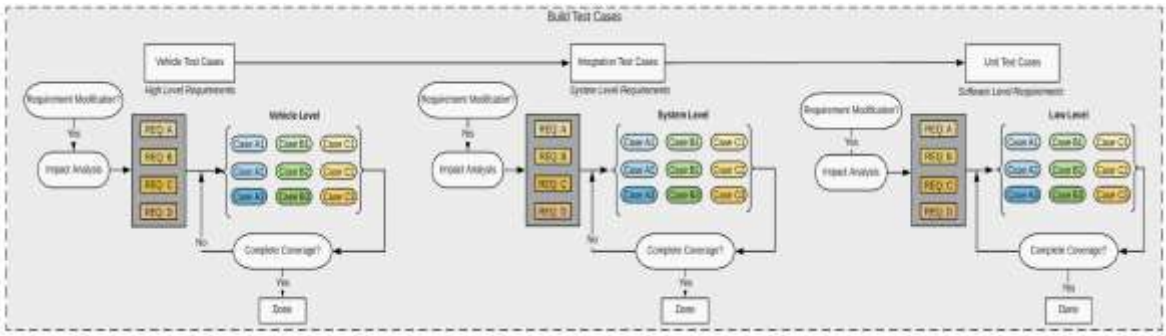

Figure 6 Complete Test Case Coverage

5. VALIDATION & VERIFICATION (3)

Validation and Verification of the model is performed from low level to high level testing. The Simulink Unit Test is performed first by implementing Unit Level Test Cases. Each requirement will be linked to a test case. Therefore, upon failed test, test cases are verified against requirements and model implementation to determine if the failure cause is a poor requirement, weak design, or incorrect test design. Once a solution has been implemented, Unit Testing occurs again to make sure the solution is valid and hasn't had any adverse

effects. After unit testing is completed, integration testing and vehicle level testing is performed in a similar fashion.

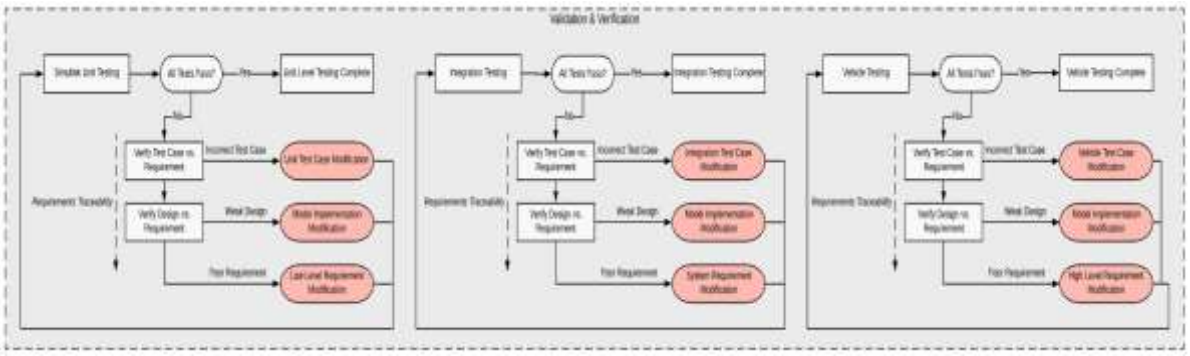

Figure 7 Complete Verification of Model

REFERENCES

[1] https://en.wikipedia.org/wiki/ISO_26262

[2] https://en.wikipedia.org/wiki/V-Model_(software_development)

[3] https://www.mathworks.com/