

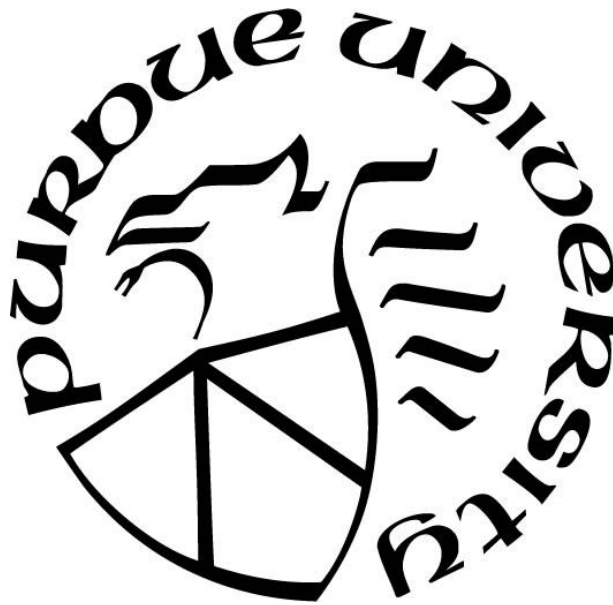
**MULTI-LEVEL REQUIREMENT MODEL AND ITS IMPLEMENTATION  
FOR MEDICAL DEVICE**

by  
**Hua Wang**

**A Thesis**

*Submitted to the Faculty of Purdue University  
In Partial Fulfillment of the Requirements for the degree of*

**Master of Science in Mechanical Engineering**



Department of Mechanical and Energy Engineering

Indianapolis, Indiana

August 2018

**THE PURDUE UNIVERSITY GRADUATE SCHOOL  
STATEMENT OF COMMITTEE APPROVAL**

Dr. Jie Chen, Co-Chair

Department of Mechanical and Energy Engineering

Dr. Shuning Li, Co-Chair

Department of Mechanical and Energy Engineering

Dr. Hamid Dalir

Department of Mechanical and Energy Engineering

**Approved by:**

Dr. Sohel Anwar

Head of the Graduate Program

## ACKNOWLEDGMENTS

There are some people that contributed to the progress of my research work who I would like to thank:

I would like to address my deepest gratitude to my advisor Dr. Shuning Li, Assistant Professor of Engineering and Technology for her kindly help about my work. Dr. Li also shared her experience about research. For this, I am always thankful.

I would like to thank Dr. Jie Chen, Professor of Mechanical Engineering and Energy and member of my thesis committee, for his helpful advices about my thesis.

I would like to thank Dr. Hamid Dalir, Associate Professor of Mechanical Engineering and member of my thesis committee, for his helpful advices about my thesis.

I would like to thank Mr. Galen Shi, and Mr. Tagore Somers, for sharing the information of auto-injector.

Finally, I would like to thank my parents for their support during my work and for giving me a chance to study abroad.

.

## TABLE OF CONTENTS

LIST OF TABLES .....	vi
LIST OF FIGURES .....	vii
ABSTRACT .....	ix
INTRODUCTION .....	1
Problem Statement .....	1
Literature Review .....	3
Requirements Engineering Models .....	4
Goal-oriented Requirement Model .....	4
Model for Requirement Engineering Processes .....	5
SysML (System Modeling Language) .....	7
Requirements Data Management .....	8
Knowledge Management .....	9
Requirements Management with Software Tools .....	9
Objectives .....	10
Organization .....	11
METHODOLOGY .....	12
Requirements Development .....	13
Goal-oriented Model to Define the Requirements .....	13
Models Using SysML .....	14
Steps on Defining Requirements .....	16
Multi-level Model .....	22
Model Implementation .....	25
Workflow .....	27
Maintenance of the Model .....	28
Excel and TC Implementations .....	29
RESULTS .....	36
Requirements Data Source .....	36
Implementation .....	38
Excel .....	38

Teamcenter Implementation .....	41
Comparison between Two Implementations.....	45
Benefits of the Multi-level Requirement Model and its Implementation .....	47
Avoiding Potential Pitfalls .....	47
Increasing the Number of Reusable Requirements.....	48
DISCUSSIONS.....	49
CONCLUSIONS.....	52
REFERENCES .....	53

## LIST OF TABLES

Table 1: An Example of Reusing Requirements with "Copy and Paste" Practice .....	3
Table 2: Examples of Different Level Requirements for an Auto-injector .....	23
Table 3: Avoiding Potential Pitfalls with New Reuse Approach .....	47
Table 4: Statistics of Requirements Number .....	48

## LIST OF FIGURES

Figure 1: Models for Requirements Engineering Process(Boehm, 2000) .....	6
Figure 2: SysML Requirement Diagrams(Michel dos Santos Soares, 2007) .....	8
Figure 3: Research Approach.....	12
Figure 4: Goal Model Fragment (Alvaro E. Arenas, 2015).....	14
Figure 5: Requirement Diagram in SysML .....	15
Figure 6: Use Case Symbol in SysML.....	16
Figure 7: Steps of Defining Requirements.....	17
Figure 8: Identify the Actor .....	17
Figure 9: Identify the Use Case .....	18
Figure 10: Refine Use Cases in Goal-oriented Thinking.....	18
Figure 11: Elicit Requirements from Use Case .....	19
Figure 12: Create Diagram.....	20
Figure 13: Choose Use Case Diagram .....	20
Figure 14: Create the Actor and Input its Name .....	21
Figure 15: Create Use Case and Input its Name .....	21
Figure 16: Create Requirement Diagram and Input its Content .....	22
Figure 17: Multi-level Sketch .....	24
Figure 18: Multi-level Requirement Model Development Workflow .....	28
Figure 19: Create the Top Part.....	31
Figure 20: Window of Part Setting .....	31
Figure 21: Input Information of Top Part .....	32
Figure 22: Create User Need.....	32
Figure 23: Input Information of User Need .....	33
Figure 24: Set the Level of User Need.....	33
Figure 25: Create Requirement .....	34
Figure 26: Input Information of Requirement .....	34
Figure 27: Set the Level of Requirement .....	35
Figure 28: Outputs of Each Step.....	36
Figure 29: Use Case and Derived Requirements .....	37

Figure 30: Part of Multi-level Requirement Model .....	38
Figure 31: Part of Template Table .....	39
Figure 32: Part of Automatically Generated Requirement Report in Excel .....	39
Figure 33: Relation Reminder.....	40
Figure 34: Partial Detail of Relation Reminder .....	40
Figure 35: Search Level 3 in Template.....	42
Figure 36: Part of the Requirements Report Automatically Generated in Teamcenter .....	43
Figure 37: Multi-level Requirement Model Development Workflow in Teamcenter .....	44
Figure 38: Requirement Change Workflow in Teamcenter.....	45
Figure 39: Percentage of Requirements Number in each Level .....	48



## ABSTRACT

Author: Wang, Hua, MSME

Institution: Purdue University

Degree Received: August 2018

Title: Multi-level Requirement Model and Its Implementation for Medical Device.

Major Professor: Jie Chen, Shuning Li

Requirements determine the expectations for a new or modified product. Requirements engineering involves defining, documentation and maintenance of requirements. The rapid improving of technologies and changing of market needs require a shorter time to market and more diversified products. As an important and complex task in product development, it is a huge work to develop new requirements for each new product from scratch. The reusability of requirements data becomes more and more important. However, with the current “copy and paste” approach, engineers have to go through the entire set of requirements (sometimes even more than one set of requirements) to identify the ones which need to be reused or updated. It takes a lot of time and highly relies on the engineers’ experiences. Software tools can only make it easier to capture and locate the requirements, but won’t be able to solve the problem of effective reuse of the existing requirement data. The overall goal of this research is to develop a new model to improve the management of requirements and make the reuse and reconfiguration of existing requirements and requirement models more efficient.

Considering the requirements data as an important part of the knowledge body of companies, we followed the knowledge categorization method to classify requirements into groups, which were called levels in the study, based on their changing frequency. There are four levels, the regulatory level, the product line level, the product level and the project level. The regulatory level is the most stable level. Requirements in this level were derived from government and industry regulations. The product line level contains the common requirements for a group of products, the product line. The third level, product level, refers to the specific requirements of the product. And the fourth and most dynamic level, the project level, is about the specific configurations of a product for a project. We chose auto-injector as the application to implement the model, since it is a relatively simple product, but its requirements cover many different categories. There are three major steps in our research approach for the project. The first is to develop requirements and classify them for

our model. The development of requirements adopts the goal-oriented model to analyze and SysML, a system modeling language, to build requirements model. And the second step is to build requirements template, connecting the solution of the problem to the information system, standalone requirements management tool or information platform. This step is to find a way to realize the multi-level model in an information system. The final step is to implement the model. We chose two software tools for the implementation, Microsoft Office Excel, a commonly used tool for generating requirements documents, and Siemens PLM suite, Teamcenter, a world leading PLM platform with a requirement module.

The results in the study include an auto-injector requirement set, a workflow for using the multi-level model, two requirements templates for implementation of the model in two different software tools, and two automatically generated requirement reports. Our model helps to define the changed part of requirements after analysis of the product change. It could avoid the pitfalls of the current way in reusing requirements.

Based on the results from this study, we can draw the following conclusions. A practical multi-level requirements management model can be used for a medical device—the auto-injector; and the model can be implemented into different software tools to support reuse of existing requirement data in creating requirement models for new product development projects. Furthermore, the workflow and guideline to support the application and maintenance of the requirement model can be successful developed and implemented. Requirement documents/reports can be automatically generated through the software tool by following the workflow. And according to our assessment, the multi-level model can improve the reusability of requirements.

# INTRODUCTION

## Problem Statement

Requirements describe how the products should be built and the functions of the products should be achieved. While requirement engineering refers to the processes of requirements being analyzed, documented and managed throughout the life cycle. It is a fundamental piece in product development. (Aybüke Aurum, 2005)

Requirement engineering is a complex task, which demands a tremendous amount of work and an enormous cost.(Doig, 2015, October 8) It begins with discovering and elicitation of requirements from all kinds of sources, then continues to test and validation of these requirements, and finally creates requirements documents. (MITRE, 2015, August 20) However, rapid developing new technologies and fast changing customer needs require the companies to provide new products to market as fast as possible and make the product diversified enough to satisfy different customers. Instead of developing a new product, reconfigurations to an existing design become more and more common. Increasing design reuse rate is a proved effective approach to reduce time to market and improve the quality of a new product. As the designs are more and more often reconfigured and reused, related requirements often get reused, too. However, how to effectively reuse existing requirement data remains an unanswered question. Some companies are using “copy and paste” to reuse the requirement documents. This is sometime a very time consuming work, since the product/system/requirement engineers need to go through the entire requirement documents to identify necessary modifications. It is also a practice which is highly rely on the experiences of the engineers. Other companies are using an information system, either a standalone requirement management tool or a module within a larger information platform (for example, a PLM or ERP platform), to management requirements data. There are two different approaches to use software tools to management requirements: manage requirement documents and manage requirements as structured items in database. Using software tools to manage requirement documents can only help to locate the right document easier and ensure that the right one is copied and pasted, but cannot avoid the reusing pitfalls. It still needs to review entire documents for reusable requirements. It will be much more effective to manage requirements as structured items within the software database. The question now becomes what the best way is to model these requirement items to

make reuse and reconfiguration of the requirement data easier, and get full benefits from the software tools.

We choose a product (an auto injector) from the medical devices industry as an application for implementation. An auto-injector is a kind of medical devices to deliver a dose of the drug for patients' self-treatment. Most of the auto-injector is spring-loaded syringes. Their syringe needles can be inserted automatically, and then the drug is delivered. We choose it for the following three reasons:

1. Requirement management is critical for medical devices industry. Food and Drug Administration (FDA) has clear and strict definition on requirements and requirement management and deliverables.
2. The auto-injector is relatively simple but refers to inter-discipline collaboration and has a lot of branches. Its structure and mechanism are uncomplicated, but its requirements cover to all different categories, including regulatory requirements, functional requirements, interface requirements and human factor requirements.
3. Our interviews with industry experts showed that more than 50% of the auto-injector product development projects were reconfigurations of existing products. Most common reasons for reconfigurations include a new drug, a particular patient population, and a new market with different regulations. The high rate of product reconfiguration makes the reuse of requirements critical for auto-injector projects.

Table 1 showed an example of how the reuse of requirement data works with the current “copy and paste” practice and the potential pitfalls with in the process.

Table 1: An Example of Reusing Requirements with "Copy and Paste" Practice

Steps #	Description of the Step	Potential Pitfalls
1	When an auto-injector is chosen as the delivery mechanism for a drug, the product engineer needs to find the requirement document of a similar product. Then copy the document for the new project.	<ul style="list-style-type: none"> <li>• Wrong product, the product may not be the one that is the closest related to the new project.</li> <li>• Right product, but wrong version of the document.</li> </ul>
2	Review the entire file to identify the reusable ones, for example, the regulatory related requirements, and the ones that need to be updated or removed, for example, requirements related to needle gauge and labeling.	<ul style="list-style-type: none"> <li>• Miss identifying reusable requirements</li> <li>• Longer time to go through the entire documents</li> </ul>
3	Revise the non-reusable ones and add new ones as needed.	<ul style="list-style-type: none"> <li>• Revised or newly added requirements contradictory or duplicate to the existing ones</li> </ul>

It is very important to develop new approaches to reuse the requirement data in order to keep up with the increasing reuses and reconfigurations in product development and avoid the potential pitfalls.

### Literature Review

Requirements define why the system is needed for every stakeholder and what the system should do to satisfy the stakeholders’ needs. The stakeholder includes anyone who is using the system or benefiting from it. All of the users, customers, and suppliers are stakeholders of a system.(Jeremy Dick. Elizabeth Hull, 2017) According to the definition from IEEE-STD-1220-1998, requirements, expressing operational, functional or design characteristic or constraint for a product or process, are derived into the natural language to make them easy to understand. (IEEE, 1998)

Requirements engineering is about the processes of development, documentation, and management of requirements.

Requirements are critical for every project. It guides the project direction and configures the steps for developing a marketable product. Since 1994, Standish group starts producing the CHAOS report, which focuses on the failure and success factors of projects. Its study shows that the incompleteness of requirements is the most critical reason for project failures, which is responsible for about 13.1% of all failures.(Standish, 2014)

Owing to the importance of requirements, it is essential to develop complete requirements and to document and manage the requirements well for a successful project.

### **Requirements Engineering Models**

Requirements Engineers adopt Model-Based Engineering (MBE) to build requirements. Model-Based Engineering (MBE) is a trend in system design. It is applied in order to define the functional specifications, systems modeling, the traceability criteria for the system, with the benefit of providing a wide communication process through a common language.(Chanaka Aluwihare, 2014)

It is an approach intends to raise the level of abstraction through the use of models in systems development activities.(Fabiola Goncalves C. Ribeiro, 2017)

Several basic models were currently used in requirement engineering focusing on developing good requirements. The goal-oriented model helps to analyze and organize the requirements logically; while the models for requirement engineering process, including the waterfall model, the spiral model, and the V and W model, are trying to optimize the development process and reduce the risk of mistake. There are also efforts to standardize the modeling language. SysML is a system modeling language. It is the most widely used to standardize and visualize the pattern of requirements. Models built based on SysML represents requirements clearly and improves the traceability of requirements.

### ***Goal-oriented Requirement Model***

In Requirements Engineering, goal-oriented requirements models are structured by answering questions: why is the system needed, what should the system do, who is the operator, and how does the operation carry out? (Alvaro E. Arenas, 2015)Then requirements come out. Goal-oriented requirements models are structured into different sub-models: a goal model which is the driving

model (the WHY), an object model to structure the domain description (the WHAT), an agent model to capture responsibilities (the WHO) and an operation model for specification level (the HOW).(Davoud Mougouei, 2016; Jennifer Horkoff, 2016; Tan Bui-Thanh, 2007)

A goal-oriented model possesses some advantages. Primarily, it will find the relationship between a system and its environment. This gives the reasons why a system is needed and its original goal. Then the process of goal specifications guide engineers to consider “why”, “how” and, “how else”. Consequently, the requirements are determined. This process generates requirements, which reduces the risk of missing or over-specifying of requirements analysis. Due to the process, goal-oriented models also allow large goals to be analyzed into elementary requirements, which are specific and realizable goals. It can reveal divergent relationships between requirements, especially the conflicts that appear because sometimes achieving one goal can interfere with achieving other goals. In addition, this model makes it possible to measure requirement completeness. By using a goal-oriented model, requirements can be discerned the completeness, their fulfillment of all the goals.(Alvaro E. Arenas, 2015; Davoud Mougouei, 2016; Jennifer Horkoff, 2016; Tan Bui-Thanh, 2007)

The goal-oriented model provides a logical thinking to analyze requirements based on the scenario of system operation. It will help us in developing the requirement data source for the project.

### ***Model for Requirement Engineering Processes***

Several approaches are used to specify processes of requirement engineering. The Waterfall model is one of the most commonly used process models, in which requirements model development moves steps by steps from requirements development, requirements documentation to implementation and maintenance of requirements, like a waterfall flowing from top to bottom.(MITRE, 2015, August 20) For auto-injector, the process starts from the development of auto-injector requirements. Then documented them for implementation in product. And engineers should maintain the requirements during the life cycle. The Spiral model is an updated version of the waterfall model. Each phase in spiral model contains design and review. Instead of a waterfall, the processes will be repeated moving in a spiral way. The repeated processes reduce the risk of the system.(Boehm, 2000)

V model consists of the system analysis, sub-model analysis and development and the system integration.(Kevin Forsberg, 2005) Following the V model, auto-injector should be refined into

sub-systems, components. Engineers develop requirements for each component and sub-system. Then requirements could be integrated for the whole auto-injector. Every time moving to the next step, verification is needed. If any problem is found, the development should go back to the last step. Because of the shape of the processes, it is called V model. W model is an improved model based on the V model. Comparing with the V model, a new process named virtual integration is added to the W model.(Christophe Ponsard, 2015; Giacomo Barbieria, 2014; Robert Darimont, 2016) The process is just like doing a simulation in software after parts design to check the assembly of parts instead of checking by real parts. The project starts with system analysis and then continues with sub-models analysis. Before detailed development, a virtual integration makes preliminary validation. If problems are found, the process should go back to analysis. Otherwise, the process moves forward to detailed development. Ultimately, system integration makes practical validation.

The following figure 1 is a demonstration of these models.

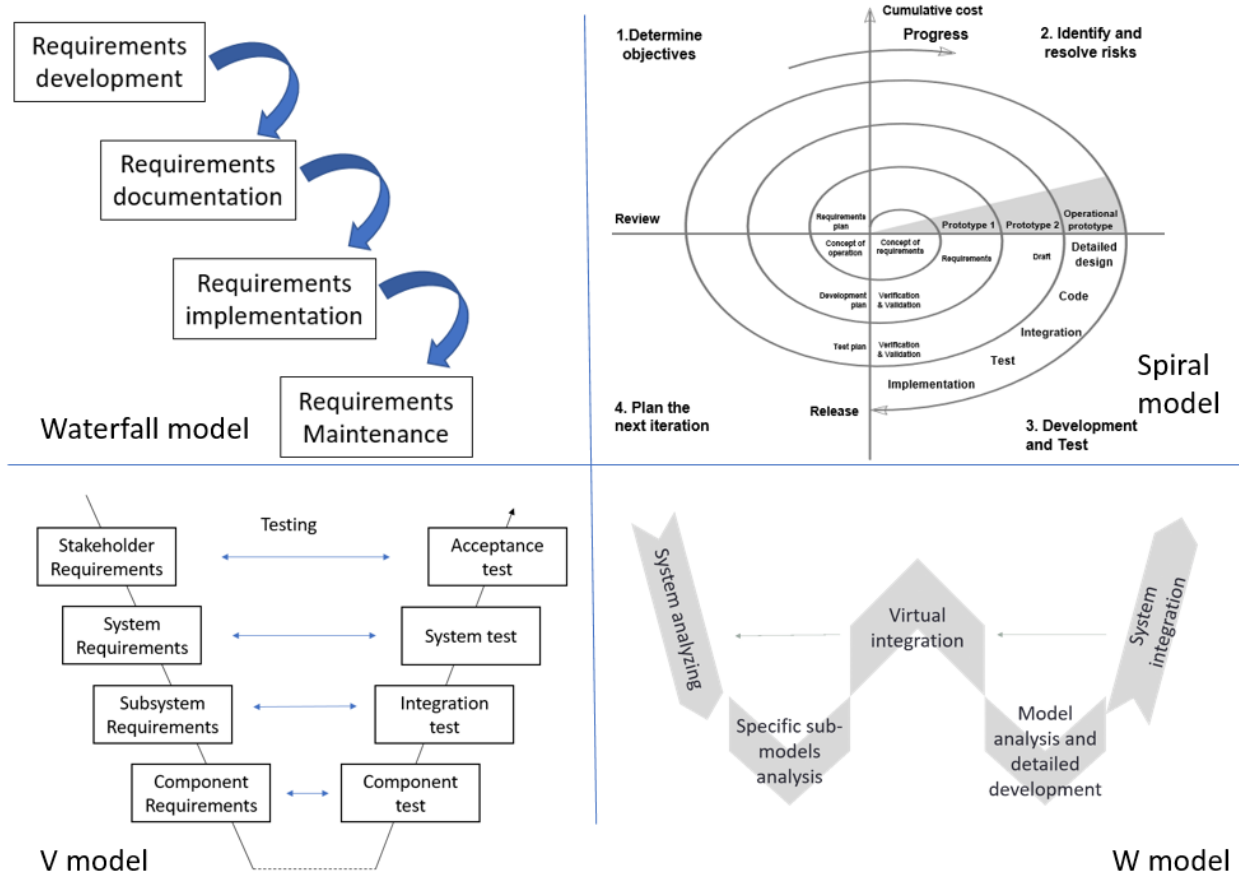


Figure 1: Models for Requirements Engineering Process(Boehm, 2000)



These models focus on the development process of requirement engineering. They are trying to reduce the risk by designing a perfect development process and to make engineers find problems before actual operation, saving time and cost.

### ***SysML (System Modeling Language)***

SysML is a general-purpose modeling language for system engineering application.(Balmelli, 2007) Researches use SysML, modeling requirements graphically, to improve specifications, organization, and traceability of requirements engineering. SysML standardizes a pattern to represent a requirement showing its all properties, in tabular or diagram. The SysML requirements diagram allows several ways to represent requirement relationships, including defining the requirements hierarchy, deriving requirements, satisfying requirements, verifying requirements and refining requirements. (Shiva Nejati, 2016) The following are parts of a requirement model named Traffic Management, which is built to manage the road traffic. For the visualization, it is based on SysML to help trace the requirements development. Figure 2 shows two SysML requirement diagrams containing their names, descriptions, and Ids. The description is the content of requirement. And the arrow shows the relationship with another requirement. These two requirements define two based functions of the traffic management system. TM12 showing in figure 2 is derived from another requirement TM13. TM 13, evolving from object management towards task and scenario management, enables TM12 that the system must provide means for expressing a wide range of tasks and scenarios, which is not only the management of traffic object, but also the interactions between traffic managers and objects. For example, the system must provide means for expressing the scenarios of controlling a set of traffic lights in an area, instead of individual lights in a crossroad. (Michel dos Santos Soares, 2007)

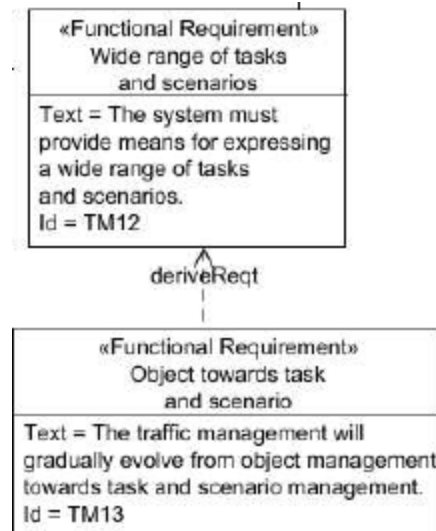


Figure 2: SysML Requirement Diagrams (Michel dos Santos Soares, 2007)

Model based on SysML is built by all requirement diagrams connecting each other with the relationships arrow. In SysML, requirements are mapped visually, properties showing in diagram and relations showing by arrows. It improves the traceability of requirements.

Using these graphical diagrams, Requirements and their relationships are mapped explicitly. It improves visualization of requirements, and at the same time helps to trace requirements. (Michel dos Santos Soares, 2007)

The SysML requirements diagrams improve the organization of requirements and, meanwhile, they can show the relationships explicitly between different requirements. There is also another advantage in using these diagrams that it helps to standardize the requirement specification due to the pre-defined semantics of SysML. (Michel dos Santos Soares, 2007; Shiva Nejati, 2016)

However, these previous efforts are all focused on developing new requirements and requirement models. They can help analyze and specify requirements well. But, there is no studies on reusing and managing requirement data to our knowledge. There is a clear lack of knowledge on how to manage the significantly increasing number of requirements and requirement models and how to reuse and reconfigure them to satisfy the fast paced changing market and customer needs.

### **Requirements Data Management**

Requirements, as a critical part of a company's knowledge body, can be managed using knowledge management methodologies, like knowledge classification. Companies can also take advantage of

software tools and information systems to manage their requirement data. Following sections will start with discussing knowledge management and knowledge classification, and followed with examining the advantages and disadvantages of the requirement management software tools.

### ***Knowledge Management***

The requirements data is an important part of knowledge to any companies. Thus, knowledge management approaches can be applied to manage requirements.

Knowledge Management is the process of creating, sharing, using and managing the knowledge and information of an organization.(John Girard, 2015) Davenport and Prusak said knowledge management is trying to make knowledge visible and show the role of knowledge in an organization, to share knowledge and offer knowledge proactively, and to build knowledge connections among people.(Maryam Alavi, 2001) The purpose of knowledge management is reducing the risk of knowledge loss and the cost of knowledge transfer under the circumstance that knowledge is stored only in the individual.

Categorizing knowledge into groups has been identified as one of the key activities in knowledge management in several studies. It is considered as an effective approach to simplify the searching, sharing and creation of knowledge. There were several studies using different criteria to classify knowledge based on their goals. Andreu & Sieber identified knowledge based on three perspectives: information, technology and the culture of firm.(Lloria, 2008) G. Hedlund's model distinguishes knowledge between three aspects of knowledge: skills, knowledge embodied in products, and well-defined services or artifacts. (Hedlund, 1994) Knowledge category models were created by Boisot and Nanoka & Takeuchi to categorize knowledge into ontological levels: individual, organizational and inter-organizational.(Lloria, 2008) However, there is no study specifically designed for classify requirements data for reuse purposes.

### ***Requirements Management with Software Tools***

Many companies start adopting software tools to manage their product related data, including requirements. The software tool can be a standalone tool just for requirement management, like IBM Rational DOORS and Dassault Reqify; but more often a module or a function within a larger information platform, which will provide a better solution to connect requirement data with the other critical product development related information. Several top information systems provide a

tool for requirement management. Siemens PLM software, Teamcenter, provides a requirements management solution which can help the companies capture and communicate requirements from every source to product decision-makers. Teamcenter could connect requirements with project plan, and make the impact of change visible. With the Teamcenter requirements management solution, requirements can be traced and maintained across organizations.(Siemens) Oracle PLM solution, Agile, also provide functions for customer need management. Agile enables users to collect all the innovation inputs, like requests, from various sources into a single sharable application. It connects engineering, design partners and sales together for collaboration. And Agile could link approved product proposals to actual items.(Oracle, 2018)

Software tools can provide support for the storing and searching of the requirement data. Many of them also help to improve the traceability and change control of the requirement data, which are essential for requirement management. However, these tools won't be able to answer the question of how to model the data and how to control and maintain the data after importing them into the tools. Significant efforts are needed to build the data model, and also define workflows and rules on how to manage the requirement data within the tools.

### **Objectives**

The Overall goal of this research is to develop a new model to improve the management of requirements and make the reuse and reconfiguration of existing requirements and requirement models more efficient. Our proposed model can help simplify the identification of the changing part of requirements by classifying them into different levels based on their changing frequency. A workflow was also developed to support the use of the model. A medical device application with two implementations was chose to verify the model and the workflow.

The specific objectives of this research were to:

- Develop a multi-level reusable requirements management model for a medical device product.
- Develop a corresponding workflow, and maintenance guidelines to support the effective use of the model.
- Implement the model and the workflow in two different software tools, an independent interface, and in a commercial platform.
- Assess the benefits of the multi-level requirement model.

## **Organization**

The remainder of this thesis contains four chapters. Chapter 2 captures the methods using in this study, which includes the development of requirement data source, the building requirement template, and the implementation of model. The application we used in study, and the way we used in building requirement model are also discussed in this chapter. Additionally, Chapter 2 stated the general workflow and the maintenance guideline for the multi-level model. Chapter 3 present the results of the methods in Chapter 2. The results contain the multi-level requirement model, the templates for two software, the specific workflow for each template, and the requirements reports generated by each template. Chapter 3 also contains the comparison of two implementations and the reusability assessment of multi-level model. Chapter 4 discusses the importance of the multi-level model, and the future work. Lastly, Chapter 5 concludes the results of this study.

## METHODOLOGY

There are three major steps in our research approach for the project. The first is to develop requirements and classify them for our model. The development of requirements adopts the goal-oriented model and use case to analyze requirements and SysML to build requirements model. And the second step is to build requirement template, connecting the multi-level requirement model to the information system. This step is to find a way to realize the multi-level model in an information system. The final step is to implement the model. We chose two software tools for the implementation, Microsoft Office Excel, a commonly used tool for generating requirements documents, and Siemens PLM suite, Teamcenter, a world leading PLM platform with a requirement module. The research approach was shown in the following figure 3.

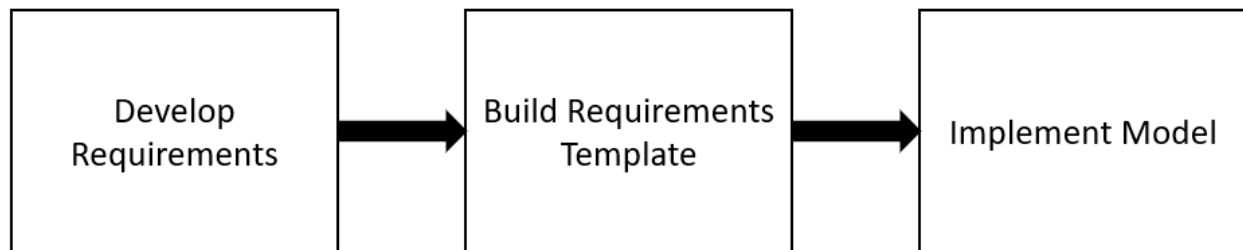


Figure 3: Research Approach

To avoid confusion, we made the following definitions which were used in the rest of this thesis.

- Requirement data (source): This is the original group of requirements developed by using requirement development process.
- Classified requirements: These are the requirements with extra level/class information.
- Requirement template: This is the base requirement model which will be implemented in a software tool. The template for different tools may have different appearance.
- Requirement model: This is the model created for a particular product development project based on the requirement template.
- Requirement report/document: This is the document containing all the requirement data for a particular project, normally an Excel or Word file. We still generate requirement

report, since this remains the most popular approach for requirement management in many companies. The reports/documents are usually used for communicating with persons/organizations outside of the information system, for example, vendors, suppliers, etc.

### **Requirements Development**

The requirement data were developed by using the goal-oriented model to define the use case, which is used to elicit requirements, and following the decomposition process to allocate the requirements. The requirements were then created by applying System Modeling Language (SysML).

#### ***Goal-oriented Model to Define the Requirements***

The goal-oriented model provides a logical thinking to analyze requirements based on the scenario of system operation. Goals may refer to functions system provided or to the quality of system needed to achieve. Agents are active components, such as humans, devices, that play some role towards goal satisfaction. Some agents thus define the software whereas the others define its environment.

In the goals' organization, goals in higher-level are integrated into general strategic, that these goals are overall and involve more agents. Relatively, goals in lower-level focus on technical details, which are more specific and involve fewer agents. That is to say, in this organization, that the development in AND way refines a goal into a set of sub-goals. In another word, AND-refinement means that it is necessary to achieve all sub-goals to make the upper goal satisfied. Whereas, in OR way, development refines a goal into a set of alternative sub-goals. OR way means that an upper goal could be satisfied by a sub-goal or another sub-goal. A requirement is an expression of a goal and how the goal could be achieved by the corresponding agent. [7-10] Figure 4 is a schematic diagram showing a part of a goal-oriented model for a business continuity plan. It is about an after-sale repair system, from the original goal contract fulfillment to the sub-goals, "repair request received", "repair request satisfied" and "repair invoice sent and paid". These are refined in AND way. To fulfill contract, all these three sub-goals have to be satisfied. The sub-goal, repair request satisfied, is refined into three alternative sub-goals in OR way, "equipment

repaired within 4 days”, “equipment repaired within 1 day” or “request no satisfied”. To achieve the sub-goal, repair request satisfied, only one alternative sub-goal is necessary to achieve.

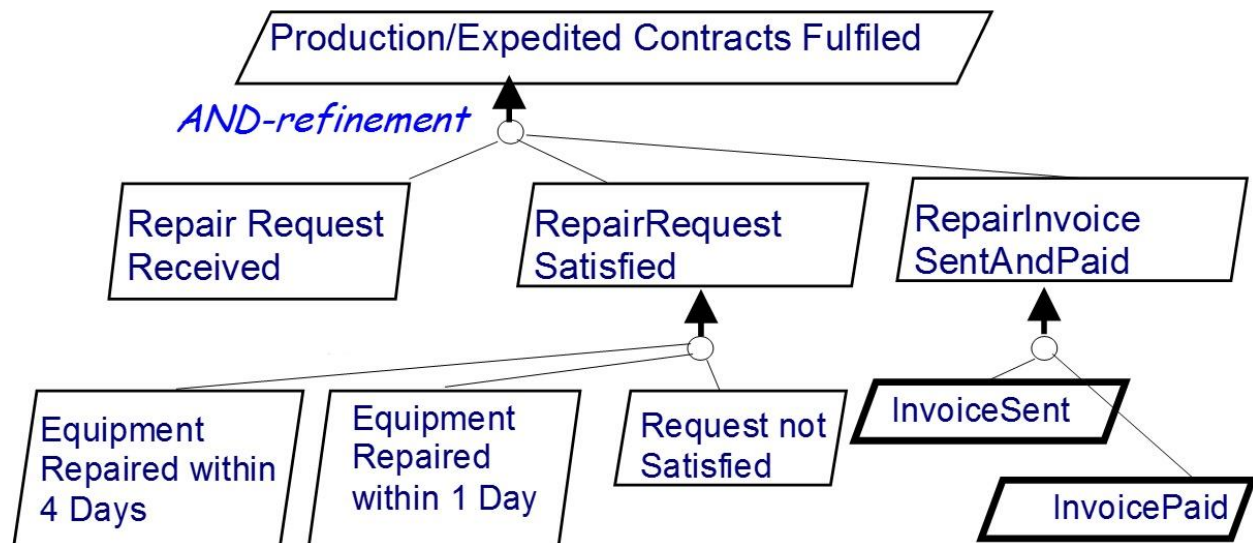


Figure 4: Goal Model Fragment (Alvaro E. Arenas, 2015)

We defined the use case by using the goal-oriented thinking. Use case, which is a kind of diagram in SysML, is used to describe the interaction between actor and the system. Regarding to satisfy the user need as a goal, several sub-goals are refined by asking how to achieve the goal. Then the goal and sub-goals could describe as actions, which are use cases. After that we elicit requirements according to the scenario describing by use cases. Detailed example are described in the following section, steps of defining requirements.

### ***Models Using SysML***

Models using SysML can represent requirements clearly and improves the traceability of requirements. It will help us in our requirements development. So we analyzed scenario of the use case in Goal-oriented thinking to elicit requirements. And our requirements are built based on SysML. These methods help to develop a good requirements model.

The SysML is a general modeling language. Requirements diagram is a unique function of SysML.(Jeremy Dick. Elizabeth Hull, 2017) As above stated, SysML standardizes a pattern to present requirements. And it also enables Use Case to help requirements elicitation. A Use Case is made of scenarios to describe the activity, which is acted by an external actor in order to achieve



a stakeholder's need. (Williams, 2004) The stakeholders' needs can be developed into sub-goals in goal-oriented thinking. With the help of SysML, not only the requirements and their relationship are visually mapped, but also their related experts are demonstrated. Therefore, we derived use cases by analysis user needs in goal-oriented thinking, elicited requirements according to the use cases, and represented requirements by SysML requirement diagrams.

A requirement in SysML is a diagram, which contains some text, and can be associated with any other diagrams. It visualizes the property of requirement. The text states each requirement, its ID, name, and description. As for the association, it refers to the relations between requirement and another element, including other requirements or use cases. The following figure 5 shows three requirement diagrams in SysML, and the bottom two requirements diagrams are derived from the upper one.

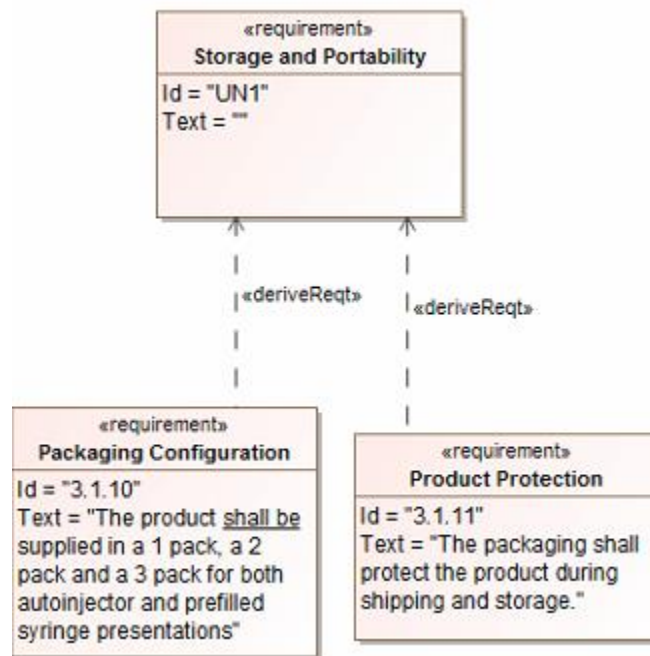


Figure 5: Requirement Diagram in SysML

In SysML, a use case is represented by an oval with its name. There is also an actor symbol, which is not a part of the system, but it interacts with the system. The actor is an agent to carry out goals. It is represented as a stickman. The following figure 6 shows part of the scenario that an actor, Delivery system, interact with a system Distribute Product. And there are three use cases. The

direct acting use case is Distribute Product which includes two use case, Packaging Configuration, and Product Protection. It is like a goal is refined into two sub-goals.

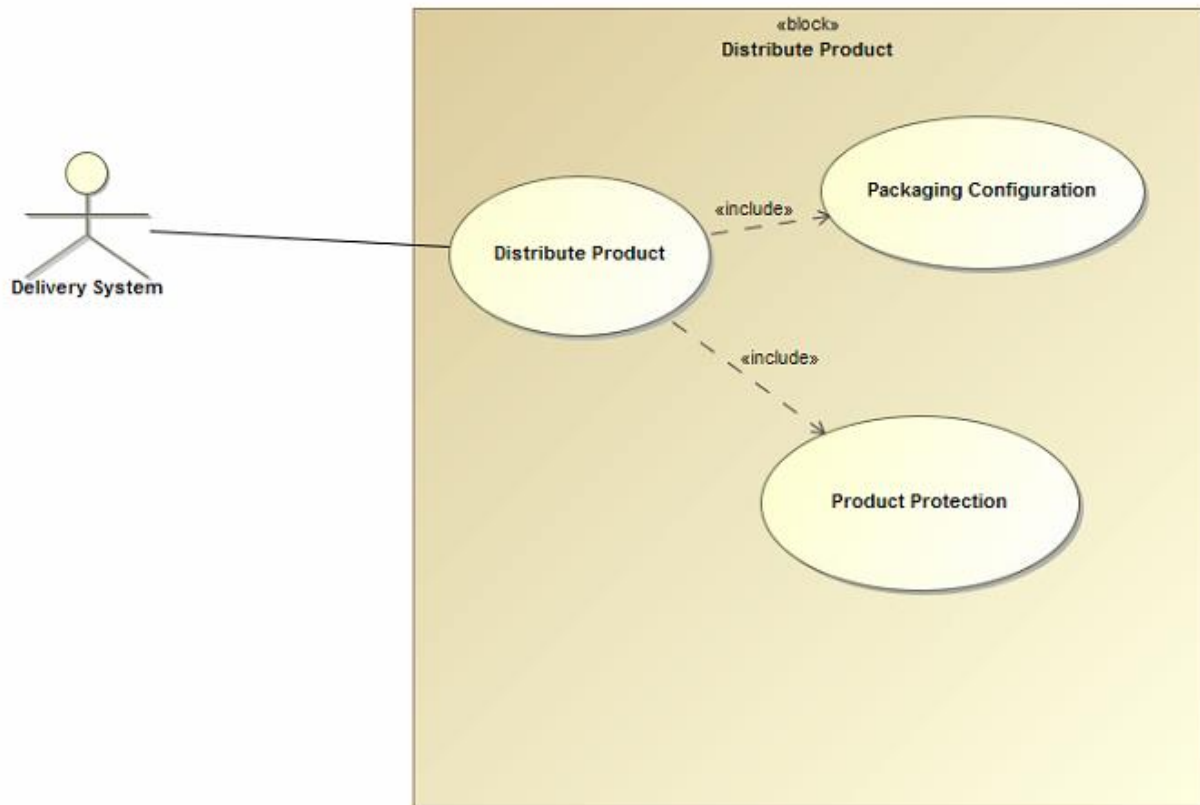


Figure 6: Use Case Symbol in SysML

### *Steps on Defining Requirements*

We developed requirements based on use case. There are three steps to define requirement. First is identifying the actor of use case. Second is identifying the use case. Then, based on the use case, we can elicit requirements. The process shows as following.

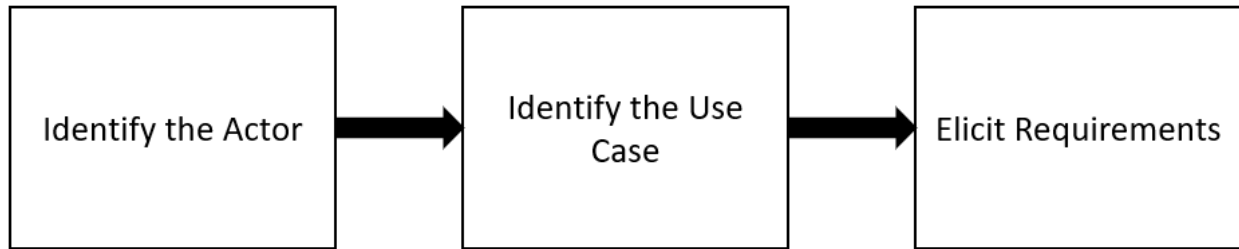


Figure 7: Steps of Defining Requirements

### 1. Identify the actor.

To start the requirements development, the first step is to identify the actors of the system. A whole set of use case model should contain every actor, which is everyone needs to interact with the system, and every use case refined to achieve the stakeholder's need for this system.(Williams, 2004)

The actors often in the answers to the following questions. Who uses the system? Who gets information from/provides information to the system?

For auto-injector, one of the user need is “safe and easy to handle”. Obviously, the actor is target patient. So we identified the actor as following figure 8.

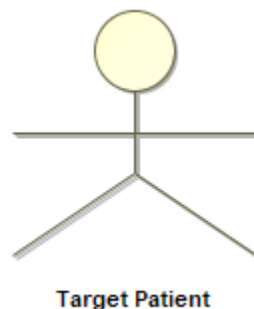


Figure 8: Identify the Actor

### 2. Identify the use case.

The development of requirements by use cases is based on scenarios. So after identifying the actors, the process comes to define the use case. The questions continue as what operation will the actor do to the system, or what functions the actor needs from the system.(Barbara Gallina, 2007; Williams, 2004) Use case could have relationships with another use case. When a use case provides information to another use case, the relationship is communication relationship, showing as an

arrow. When a use case includes other functionalities, the relationship between use cases includes relationship. In SysML, the include relationship is represented by an arrow and labeled with <<include>>.

What operation will the target patient do to auto-injector? Then we can define the interaction between target patient and auto-injector as “handle product”. The use case shows in figure 9.

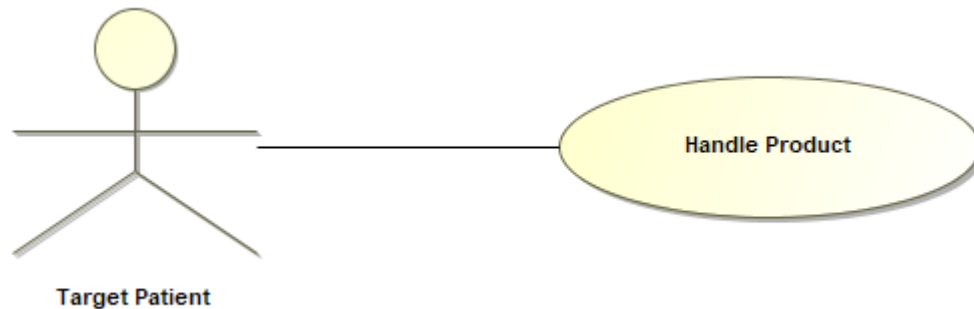


Figure 9: Identify the Use Case

While identifying the refined use case, we analyze in the goal-oriented thinking. In order to achieve the user need “safe and easy to handle”, product has to be used effectively and be handled safely. We can refine two sub-goals in AND way, “to use effectively” and “to minimize use error”. Then we can transfer these goals into interactions. So we can get that the use case “handle product” include two use cases, “use effectively” and “minimize use error”, showing in the figure 10.

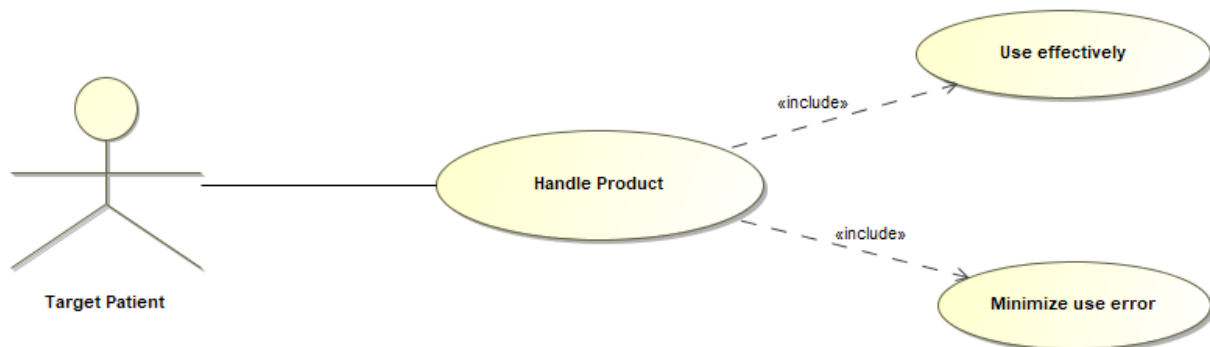


Figure 10: Refine Use Cases in Goal-oriented Thinking

### 3. Elicit Requirements.

When it is done with identifying use case, represent it in SysML symbols to visualize the system. And it will be easier to understand what will happen and to derive a textual sequence of the use case. The sequence is so-called flow-of-event. The flow-of-event is a description of what the system should do, and how the system does it. Then requirements come out by the expression of the event in natural language.

According to the refined use cases, we can derive two child requirements from the user need “safe and easy to handle”, showing in the figure 11. One is that the forces and torques required to operate the product shall enable safe and effective use. Another is that the product shall minimize or eliminate the likelihood of use errors for target patient as determined by risk management.

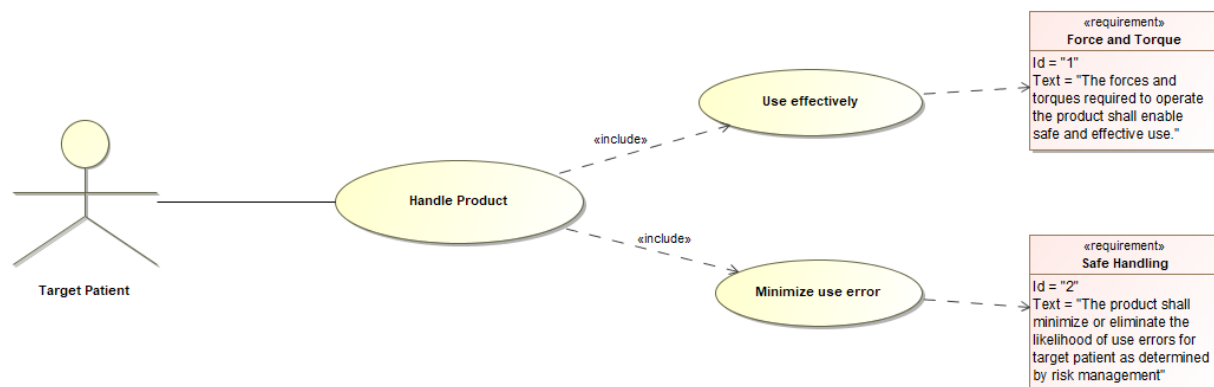


Figure 11: Elicit Requirements from Use Case

We process the above steps in Cameo System Modeler. The Cameo System Modeler, which provides the environment of SysML, was chosen to build our requirements. Cameo Systems Modeler is an industry software providing Model-Based Systems Engineering (MBSE) environment to define, track, and visualize all aspects of systems in the most standard-compliant SysML models and diagrams.

The detailed steps demonstrate as following.

1. Create diagram.

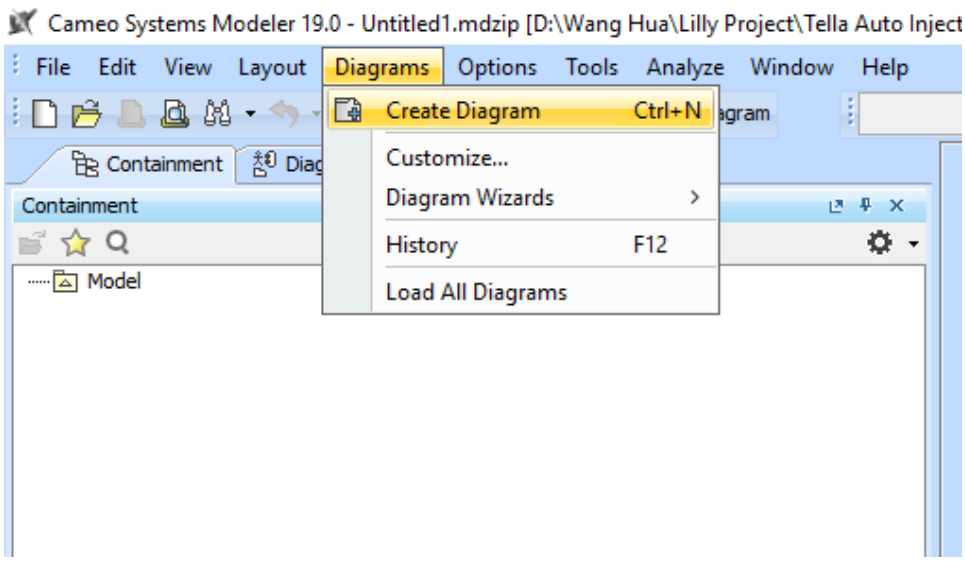


Figure 12: Create Diagram

2. Choose use case diagram.

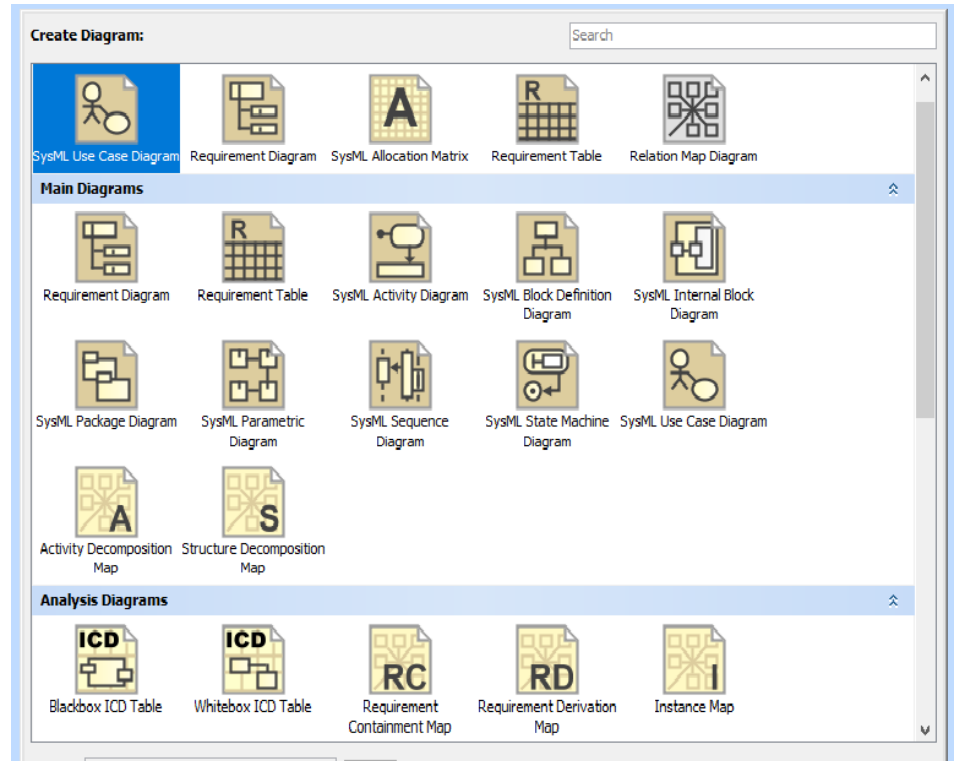


Figure 13: Choose Use Case Diagram

3. Create the actor and input its name.

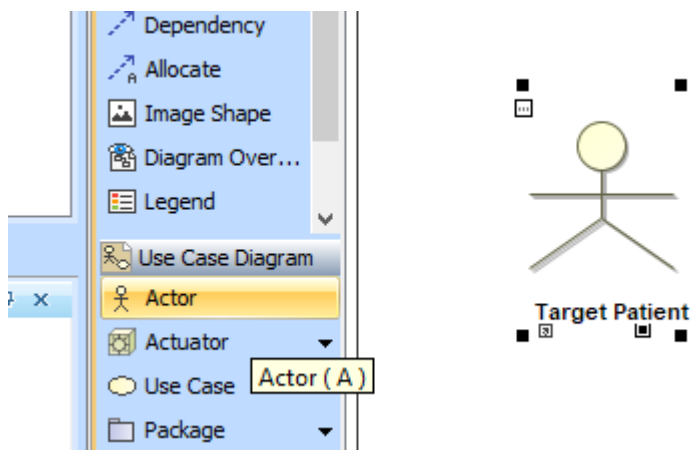


Figure 14: Create the Actor and Input its Name

4. Create the use case and input its name.

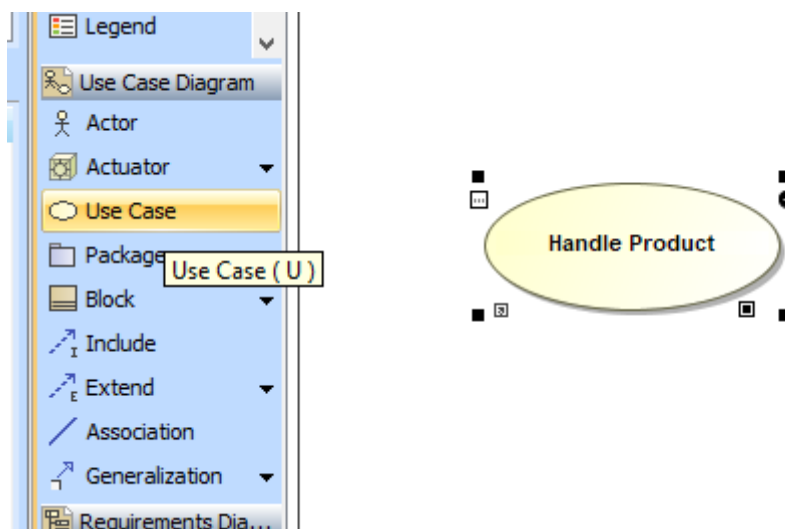


Figure 15: Create Use Case and Input its Name

5. Create requirement diagram and input its content.

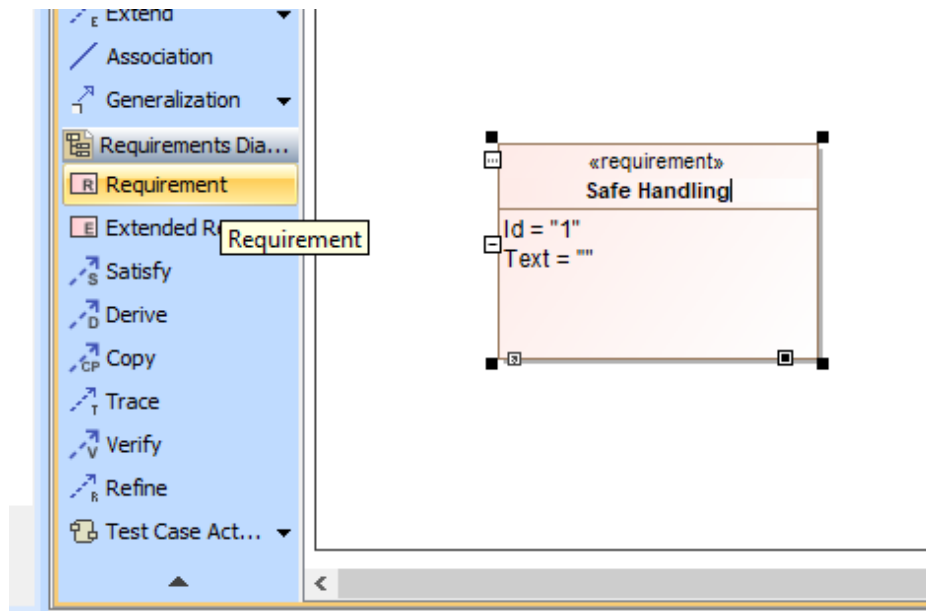


Figure 16: Create Requirement Diagram and Input its Content

### Multi-level Model

Requirement classification is a helpful way to manage requirement data. A clearly defined classification schema is critical to the success of requirement classification and management. Considering the reusability of requirements model as the main goal, we categorized requirements into groups based on their changing frequency.

There are four different groups in our classification schema, called four requirement levels: the regulatory level, the product line level, the product level, and the project level.

The regulatory level contains the most stable part of requirements. The product line level is about the basic requirement for main features of a product family. The product level contains more specific requirements for a single product. The project level refers to the requirements of configuration. The first level, we use the government and industry regulations as the foundation. Requirements in this level set the basic necessity for the product design, which is the most stable for the requirement model. It is unnecessary for the basic level to be updated in every project. Next level is the product line requirements. It is a set of products, which have common features. These requirements in this level are set to define the basic functions that the product family should provide. Table 2 showed examples of different level requirements for an Auto Injector.



Table 2: Examples of Different Level Requirements for an Auto-injector

Level	Example Requirement
Regulatory	ID: 1.18.1, Name: Specified Requirement Servicing, Text: the FDA stipulates that each manufacturer shall establish and maintain instructions and procedures for performing and verifying that the servicing meets the specified requirements, where servicing is a specified requirement.
Product Line	ID: 2.7.1, Name: Auto-Injection, Text: The device shall deliver the dose by auto-injection
Product	ID: 3.7.4, Name: Container Type, Text: The Delivery System shall be designed to function with a pre-filled 2.25ml round flange syringe with Rigid Needle Shield (RNS) as its container closure.
Project	ID: 4.7.5, Name: Needle Gauge, Text: The Delivery System Container Closure shall use a 27-gauge thin wall or smaller needle.

Figure 17 is a structural demonstration of the multi-level requirements model. We number the ID as following rules. The first number is requirement's level. The second number is the number of user needs which the requirement derived from. The latter numbers describe the location of the requirement in the development process. Different levels of abstraction allow a separation of concerns among domains of experts. The specific group of users has access to the corresponding level of requirements. The manager can access the high level, but the parts designers only have low-level access. When the change happens, it is easy to determine the affected contents. By multi-level model, the corresponding part of knowledge in requirements engineering is connected to certain people. And the role of knowledge will be defined when the requirements are classified into levels.

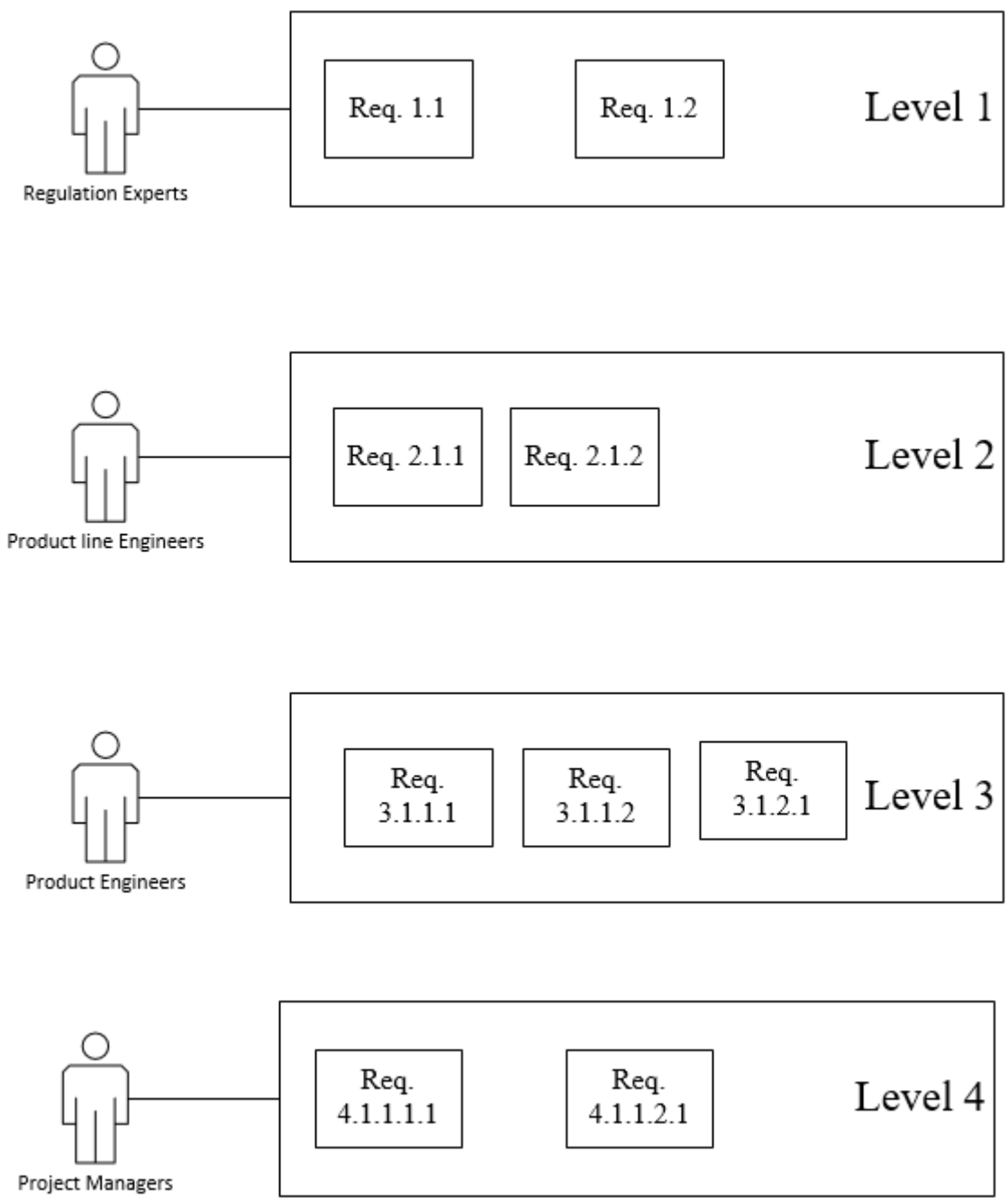


Figure 17: Multi-level Sketch

The multi-level requirements model is just like to assemble requirements as the parts assembled, from fundament to alternative branches. As a result, it makes the recombination to build a new requirement model easier, disassembling alternative requirements and adding new requirements, when product reconfiguration happens. Similarly, an instance for the reconfiguration, Arvid

Butting and his group in 2016 did a modeling for reusable robot assembly processes. They divided the assembly into the process level, task level, and skill level. Using skills fulfills the task, and tasks go through the processes. Different levels are designed by different domains experts. (Arvid Butting, 2016) The modeling can still work well with the same skill level when the process changes. Thus, same as our multi-level requirement model, different level refers to the different part corresponding to specific domain experts. Similar to the car models, manufacturers cannot design a new engine for every new model. The styling varies, while the engine remains. Therefore, we make the core of our model stable and make the lower levels changeable. Our requirement model classifies the requirements from high-level industry regulation to the lowest level configuration requirements of a specific version of the product.

There were three main advantages of the multi-level model.

1. In order to reuse, the multi-level requirements template for a product will be stored. So the most of requirement knowledge for this product, will be stored in the organization. The requirements data could be shared with other people who have access. The knowledge won't be lost even if the related experts leave.
2. Multi-level model helps to locate the corresponding groups of requirements when a change happens. Rebuilding work of requirements can reuse requirements by groups. In consequence, a multi-level requirements model is demanding to improve the reusability of requirements model and to help in knowledge management.
3. There is also another advantage of multi-level requirement model. Under normal condition, it makes the requirements easy to build, manage and maintain. During the building of the requirements model, level classification helps analysis. Building the model with derived requirements from high-level to low-level increases the completeness and reduces the risk of missing requirements.

### **Model Implementation**

The model was then implemented into two different software tools, Microsoft Office, Excel and Siemens PLM, Teamcenter. There are two main reasons for the implementation: first, it can help to test the feasibility of the model, and second, a good implementation strategy is critical to the success of the model. Without a good implementation, the stakeholders won't be able to see the advantages of the model and may cause more errors and confusions in the future. We want to use

the two implementations in these two different software tools to provide guidelines and best practices on implementation strategy. The choice of the two software tools were carefully considered. These two tools represented two different requirement management approaches and two different information system mature levels. Excel is the currently most commonly used tool for requirement documentation. The implementation with Excel can be used as a standalone tool, but also can easily be adapted as a middleware or an interface to another information system or software tool. While Teamcenter represented the highest level of integrated requirement management within an enterprise level information system with much more sophisticated data structures and functions. The Teamcenter implementation will provide an example of using the model in a complexed information environment and take full advantage of the capability of the information platform.

There are several challenges and important questions in the implementation of the requirement management model.

1. How to align the multi-level management model with the requirement model built based on requirement decomposition?

Requirement decomposition provides a hierarchy view of the requirements, and also makes the tractability between different requirements become clearer and more intuitive. It is the most common method to organize requirements. We want to remain this top down tree view of the requirements to make it easier for the engineers to accept our implementation, but also want to include the level information to support the reuse of the requirements. We end up choose two different approaches for the two implementations. For Excel implementation, we add an extra digit in the first of requirements' coding schema to represent the level information. While for Teamcenter implementation, we add the level information as a new attribute to each requirement items. This new attribute can be used as one of the querying criteria and also can be used for access control and configuration management.

2. How to create new requirement models after the implementation?

A workflow is needed to provide guidelines on how to use the requirement templates after they were built in the software tool. This workflow should cover different product development scenarios, and need to be modified for different software tools.

3. How to maintain the requirement template to ensure its accuracy and validity?

The requirement template is not a static item, but a dynamic one. This means that the template needs to be understood through continuous reviews and updates. An obsolete template will not bring any benefits, but rather cause troubles and even disasters. There has to be clearly defined and documented processes and guidelines on the maintenance of the template.

The details on the two implementations and how these challenges were addressed will be discussed in the following sub-sections.

## **Workflow**

For the reusability of the requirement model, we need to take advantage of the existing template and minimize the changes. Therefore, it is necessary to analyze the change of product first, to determine which part of requirements is needed to change before moving on to developing the requirement model for the new product.

Based on an original requirements template, when a new product design is presented, the workflow becomes as follows. We can only change the configuration part of the requirements if the new product is a reconfiguration of the original one. The requirements model has to change all the corresponding requirements of this type of product if the new product is a new category product. If the new product belongs to another field, it has to rebuild a whole requirements model. Figure 18 shows the workflow of the change of requirements. When changes happen, the first is to define which part is needed to change. Then keep unaltered levels and remove undesired levels. After that, process and complete the new requirements model. Instead of making a fresh new requirements model, the requirements can be changed simply if the reconfiguration happens. In the workflow, the first step is defining the change of the product. If it is a new field of product, the requirements should start with the regulatory to build a whole requirements model. If the product is a new category but in the same field, the industry regulations are the same, but it needs new product line requirements. So the regulatory requirements remain and the steps continue with the specification of level 2 (product line requirements), and then move to lower level until finish the whole requirements system. If the product is just a new configuration, the steps directly start with the specification of configuration requirements, keeping the level 1, 2 & 3 (regulatory, product line and product requirements).

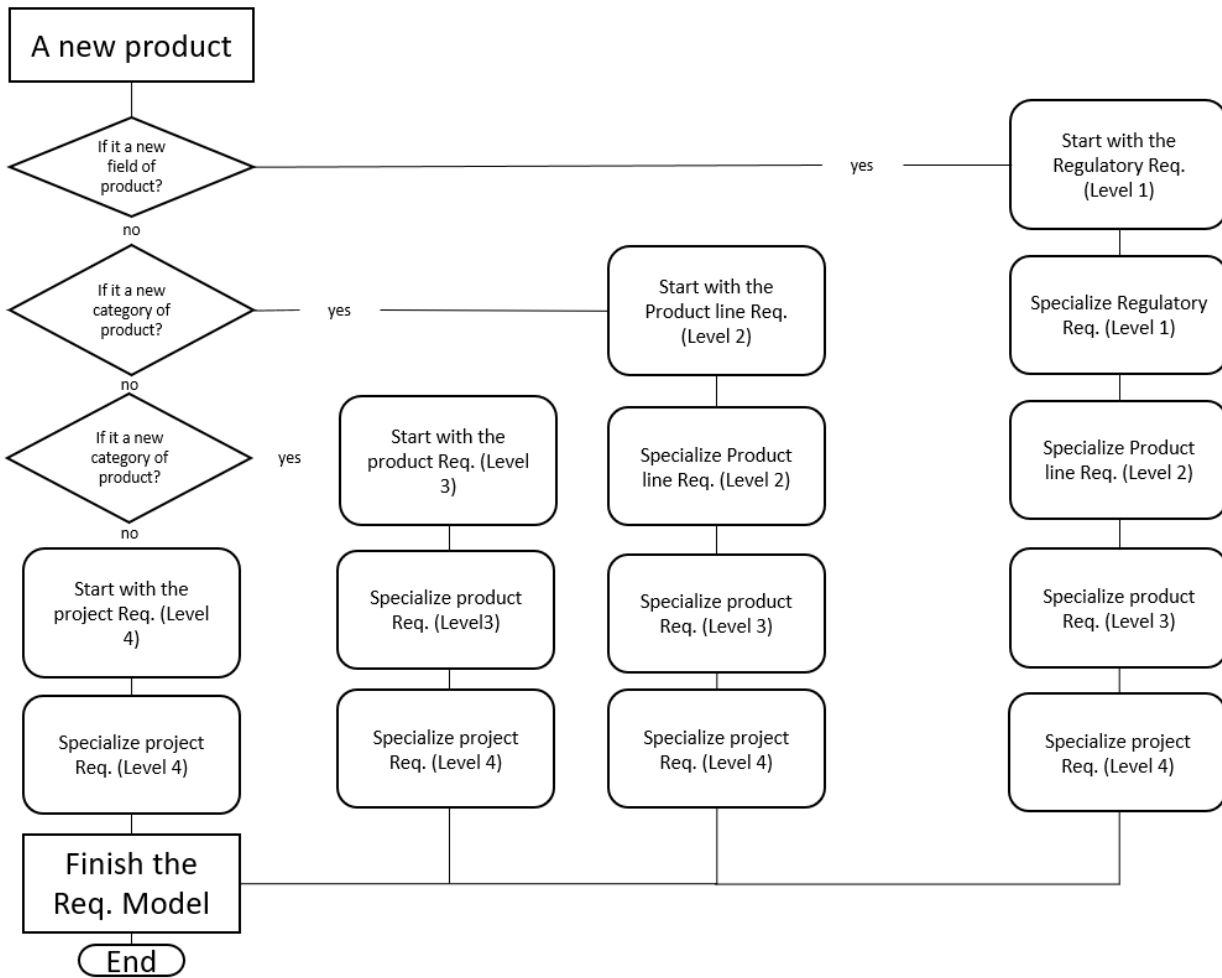


Figure 18: Multi-level Requirement Model Development Workflow

### Maintenance of the Model

After releasing a requirements model for a project, the following job of requirements engineering is the maintenance of this multi-level requirements model. It is critical to making the model dynamic, in order to keep it most suitable for the current projects. There are two kinds of maintenance, regular maintenance, and need-based maintenance.

The first is regular maintenance, which is periodically reviewing all the requirements of the template and the changes happened in previous projects. The purpose of the reviewing is to define if there are any changes significant enough to change the template permanently. For instance, some requirements may be not necessary anymore. They will be deleted after maintenance. And there may be some new stable requirements appearing after a period of time. So that template will add

new requirements during maintenance. Or some critical information of requirements has to change permanently. In a word, the regular maintenance is reviewing documented information, based on the frequency of changes, to update the template.

To execute the permanent change, every change happening for every project has to be documented. And the collection of these changes will be reviewed by a specific group of experts regularly. Then, experts will define if the change is stable enough to be changed in the template. Therefore, according to the assessment of experts' review, the requirements template has to be updated periodically.

The second maintenance is based on the need. It will be triggered once a significant event happens, for example, an FDA regulation change or a major recall. Similar team of experts need to be included in the review and change control meetings to ensure the changes are addressed properly and the templates are updated correspondingly.

### **Excel and TC Implementations**

We chose Excel as one of the tools to implement the multi-level model. There are two main reasons for this choice. First, it is a very widely used tool and many companies are currently using it for requirement documents. An implementation in Excel will be easier for engineers, without much training, to understand and use. Second, an implementation in Excel can also be used as a middleware to interface or integrate into other information systems.

Besides, Macro in Excel enables us to program functions for the convenience of implementation. In the Excel table, the same level's requirements can be listed in one section, in order to make the levels of requirements obvious to be located. The checkbox in Excel can be linked with each requirement in the table to make the requirements selectable by click. The more powerful capability is that the checkboxes and buttons inserted in the Excel sheet could embed code. These encoded items enable kinds of operations on this implementation.

The most important function of the code is to store the relationships among requirements by the programs of the linked checkboxes, though there are only requirements listed in Excel sheet. And because of the stored relationships, reminder shows when engineer selects related requirement that has a specific relationship with others. And associated requirements can be selected automatically. On the contrary, the conflicted requirements will be not selectable, showing with another changed background color in the row of the related requirement. By distinguishing the checkbox value, and

storing the related rows of the true value checkbox, the encoded button also makes it possible to generate a requirements report after engineer finishing the selection of all desired requirements for the new project. Sometimes engineering may want to clear all selected requirements while mistake happening. This function is realized by changing values of all checkboxes to “False”. The opposite value “True” means the requirement is selected. Furthermore, code enables scrolling screen to a certain row, which makes engineer can locate desired levels quickly.

Teamcenter is another tool we adopted to implement our model. Teamcenter is a suite of product lifecycle management (PLM) computer software applications. It is one of the most widely used commercial software for product lifecycle management. We implemented our requirements model by using the product structure module in Teamcenter. This implementation showed the feasibility and advantages of using our model together with an information system. We chose the product structure module for our implementation since it is one of the core PLM capabilities that all major commercial platforms provide (maybe under a different name). This makes our implementation more representative and can be easily adapted for other PLM platforms.

Teamcenter enables items created as product structure. Product structure is a hierarchical decomposition of a product, typically known as the bill of materials.(Wikipedia, 2018, February 10) It is a tree structured breakdown of product. In the structure, sub-items are assembled under the parent item. It is like requirements organized in development process relations from parent requirement to derived requirements. The following is the process of creating requirements template in Teamcenter.

To build the requirement template in product structure, the operations start from creating new item, to choosing the type of item, to setting the attributes of the item in the dialog windows. Since we are building a product structure, we use part as the type of the new item. First is to create a top level item, which is the requirement model for auto-injector. Then, user needs are created under the top item. And the building of requirements continues following requirements decomposition. The location of new creating item is under the selected item while clicking the new button. Select the parent requirement and create new requirement. Then the new requirement will be under the parent one. Derived requirements created under each user needs, and sub-requirements follow. The most important work is adding the level as an attribute for each item, according to the classification of requirements. In Teamcenter, we set the level attribute of requirements as a reference designator. And filter can search the requirements in the template by the designator so that certain levels of



requirements can be easily found. Reference designator is a reference to identify a component in schematic. Filter in product structure management can search item by designator. So we choose the designator as the level attribute, in order to enable searching entire level's requirements.

The detailed steps process as the following.

### 1. Create new part, the multi-level model for auto-injector.

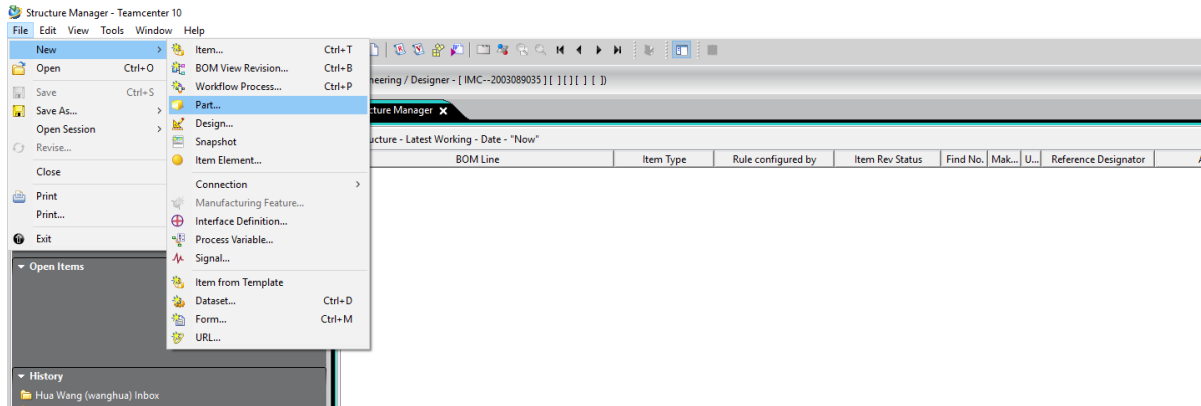


Figure 19: Create the Top Part

### 2. Appear a window for part information setting, click next.

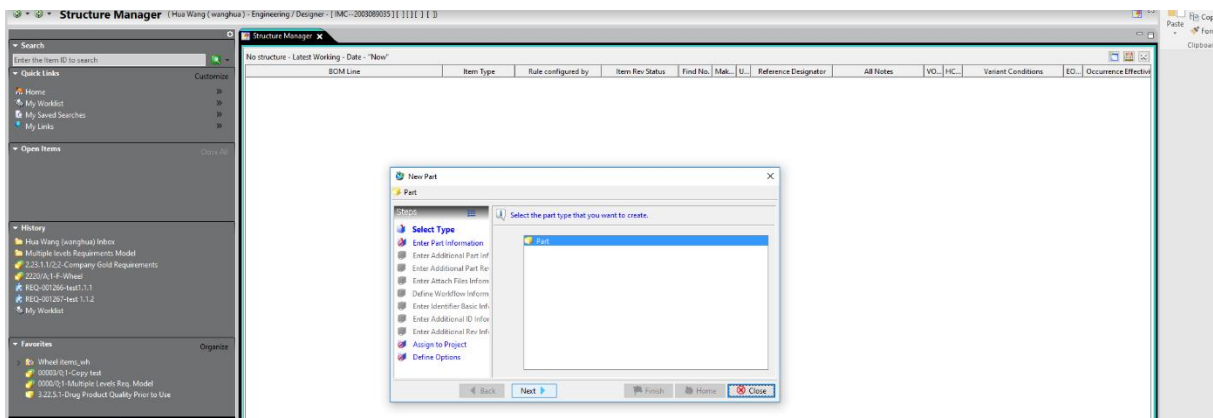


Figure 20: Window of Part Setting

- Input the ID, Revision, Name and Description. (we didn't revise, so the revisions for all requirements are setting as 1.), then click finish.

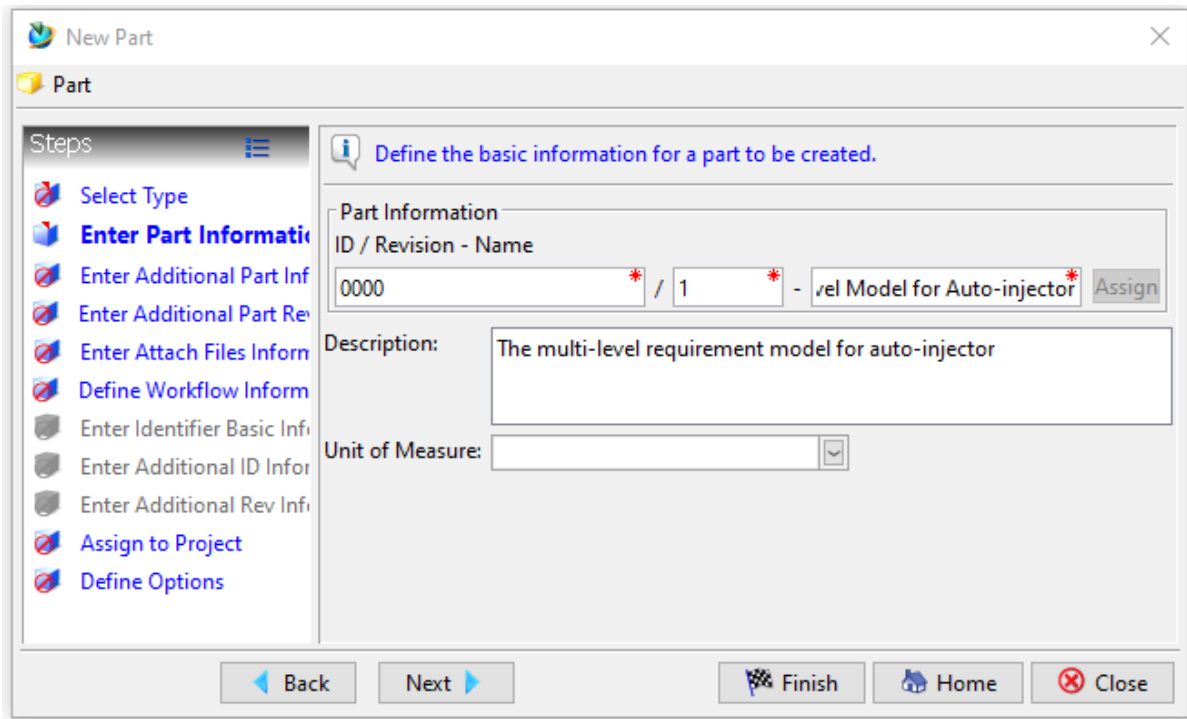


Figure 21: Input Information of Top Part

- Create user needs under the top part, while the upper part is selected.

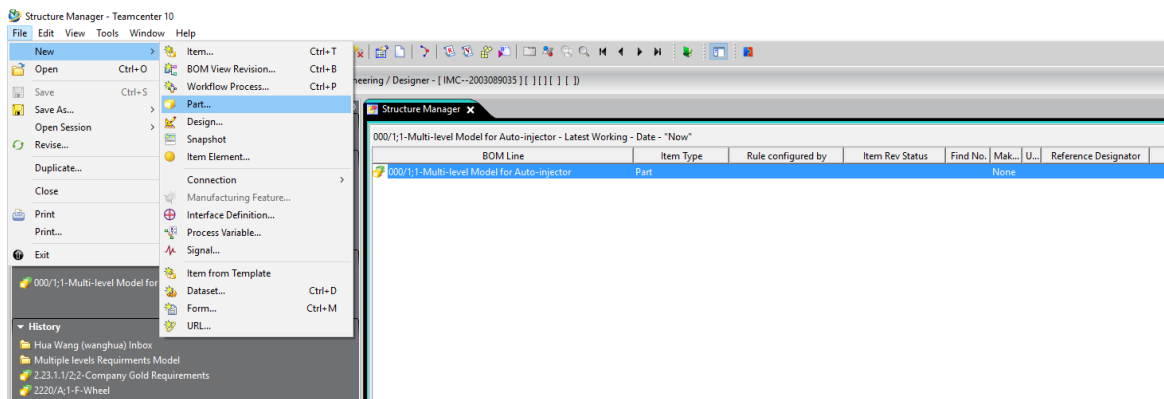


Figure 22: Create User Need

5. Input the information of this user need.

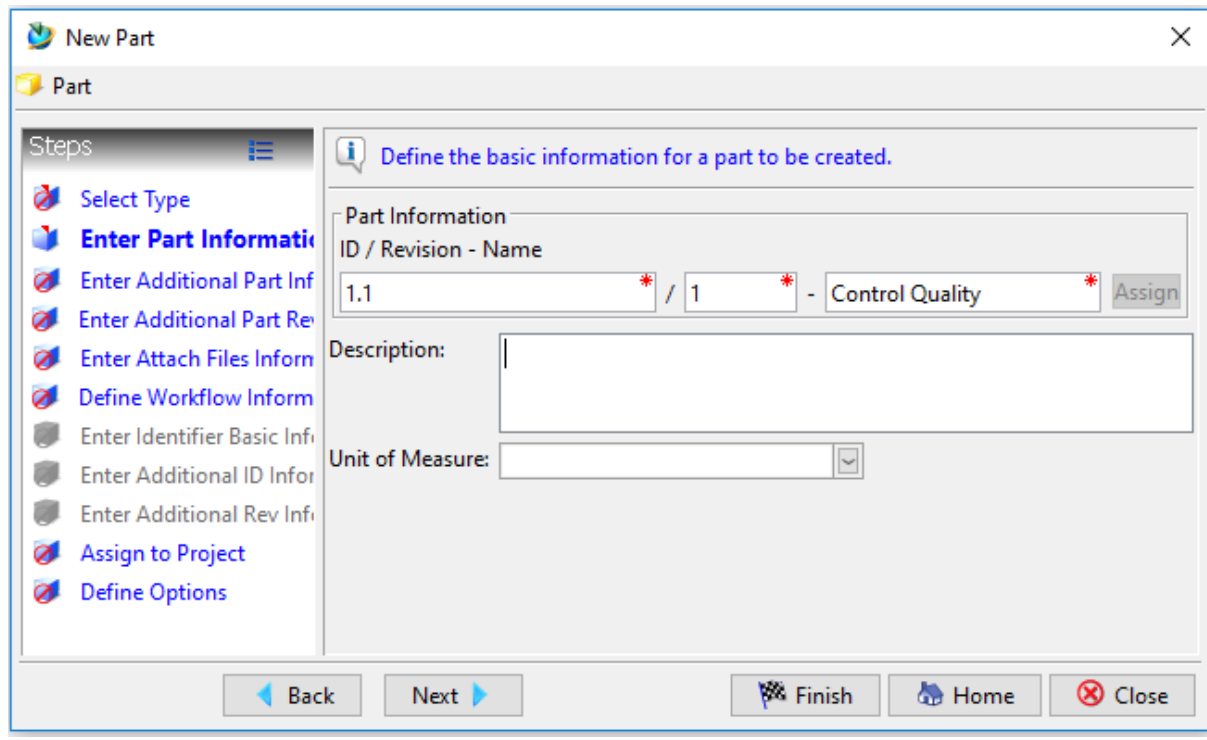


Figure 23: Input Information of User Need

6. Set its level as reference designator.

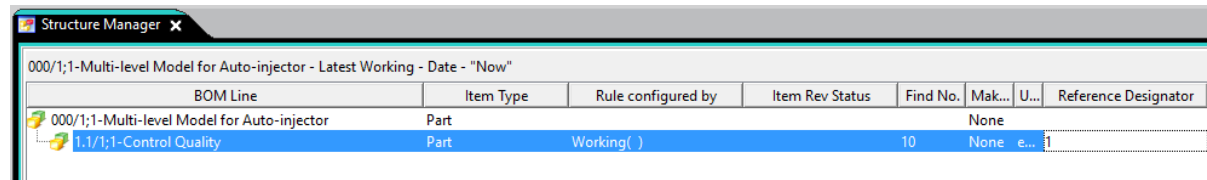


Figure 24: Set the Level of User Need

## 7. Create derived requirement under user need.

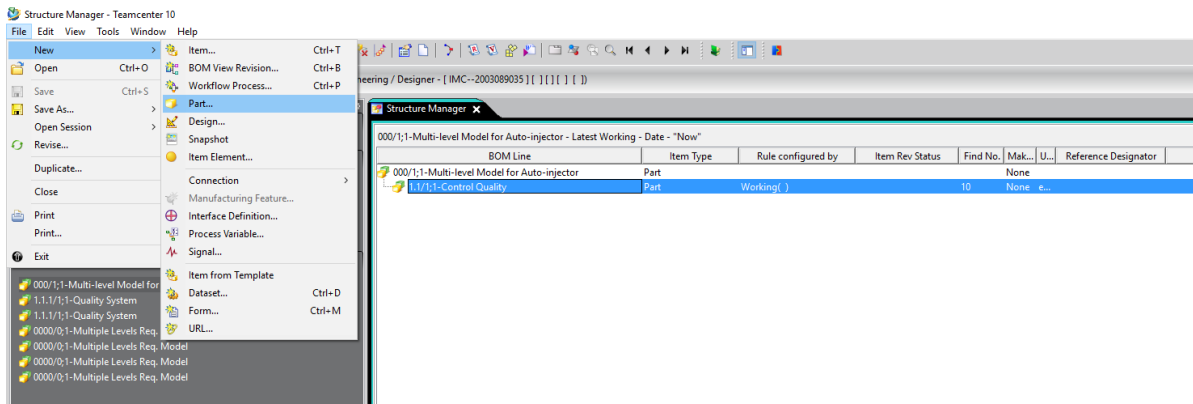


Figure 25: Create Requirement

## 8. Input information for requirement.

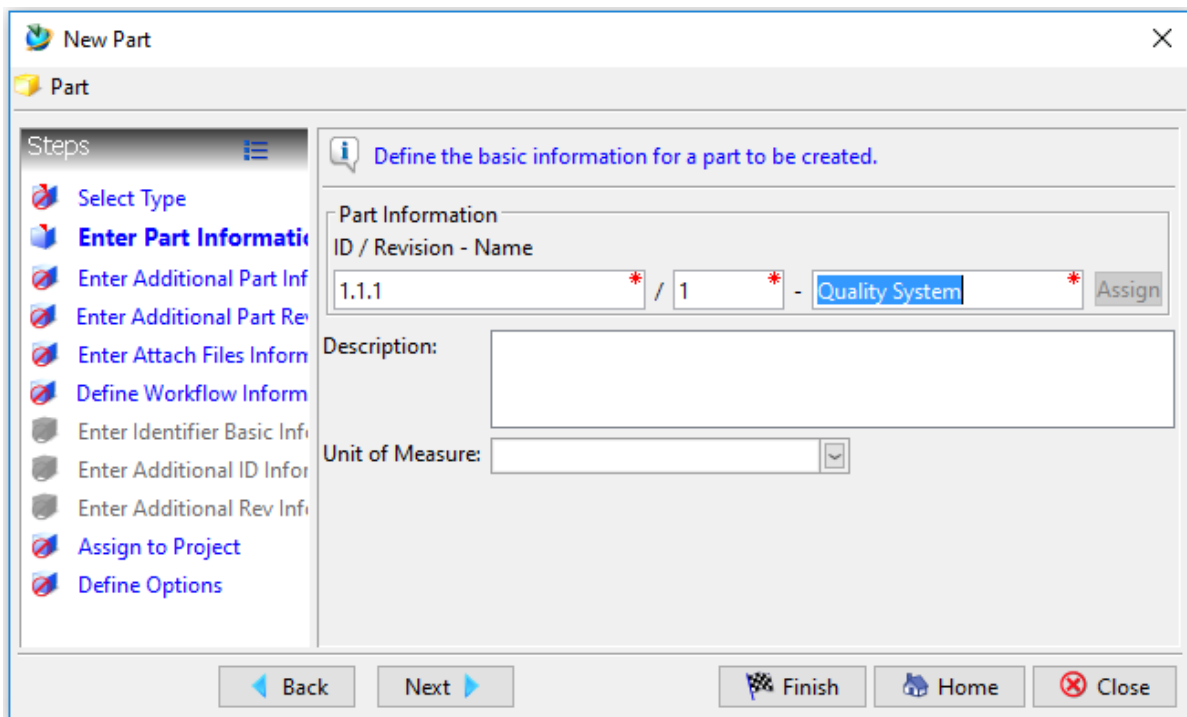


Figure 26: Input Information of Requirement

9. Set its level as reference designator.

BOM Line	Item Type	Rule configured by	Item Rev Status	Find No.	Mak...	U...	Reference Designator
000/1;1-Multi-level Model for Auto-injector	Part				None		
1.1/1;1-Control Quality	Part	Working( )		10	None	e... 1	
1.1.1/1;1-Quality System	Part	Working( )		10	None	e... 1	

Figure 27: Set the Level of Requirement

The building work continues to repeat above steps for the rest of requirements.

## RESULTS

The results from each major step are shown in Figure 28. From the first step, requirements development generated requirements data source, which is classified into levels according to their stability. These requirements are then used in the second step to build requirements template for an information system. And the final step outputs specific workflow of each implementation tools and requirements reports.

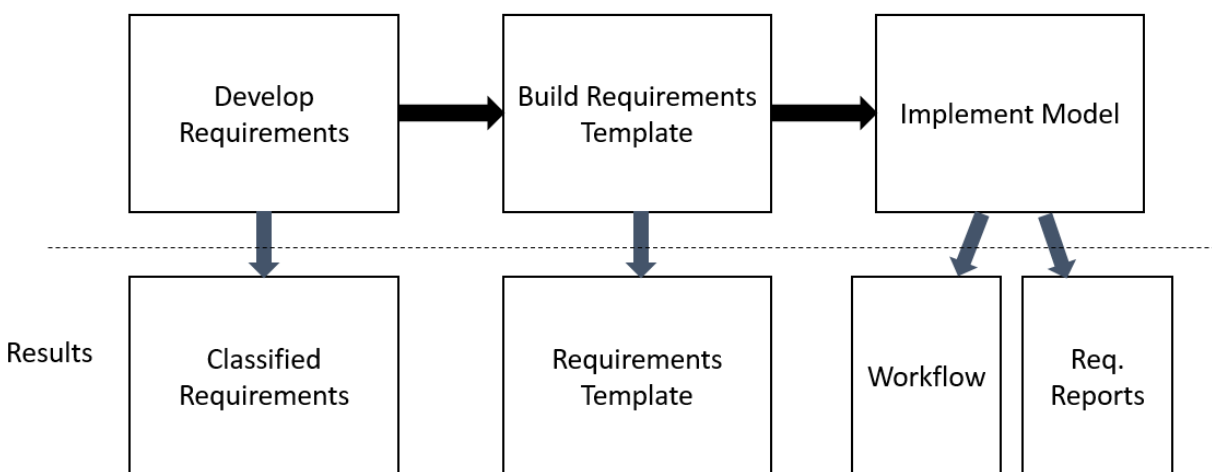


Figure 28: Outputs of Each Step

### Requirements Data Source

After setting application for implementation of multiple level model, the first step of research starts. And the first step outputs the requirements data source. These requirements, classified into levels, are developed for auto-injector.

During the building process, we use the Cameo System Modeler, which is a software of Model-Based Systems Engineering (MBSE) environment, to make the development of our Requirements Template easy. It helps to analyze and abstract in the development of requirements by its graphical use case and requirements diagrams.

A use case can show the development process from user needs to their corresponding requirements. In the use case, it graphically describes what activities the system should do by which actor to satisfy the user needs from stakeholders. It is also a microscopic demonstration of corresponding

relations between requirements and experts. Requirements can derive from these use case. Figure 29 is a sample of the use case and the requirements derived from it. The user need is protecting safety during the delivering of product. So the actor is delivery. to deliver product. Before customer get product, the delivery has to distribute product and store product. So there are two use case for protecting safety during distribution. And based on the use case, two requirements are derived from the user need. The requirement “drug product quality during distribution” was elicited by the use case “distribute product”. The requirement “shelf life” was elicited by the use case “store product”.

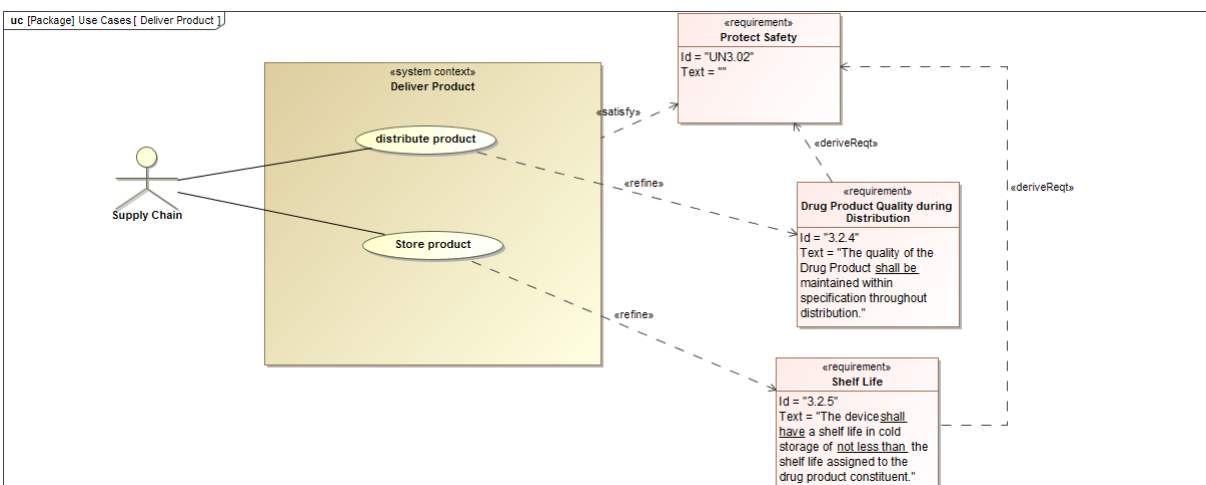


Figure 29: Use Case and Derived Requirements

The following figure 30 shows part of the multi-level requirement model, which are documented in the table. They are already classified into four levels.

ID	Requirement/Spec Name	Functional Requirement	Requirement/Spec Description	Specific Value	Requirement/Spec Category	Rationale	Stakeholder	Functions	User Needs
Level 1									
1.1.1	Quality System		Each manufacturer shall establish and maintain a quality system that is					Quality System Control	Control Quality
1.1.2	Quality Policy		Management with executive responsibility shall establish its policy and objectives for,					Quality System Control	Control Quality
1.1.3	Organization		Each manufacturer shall establish and maintain an adequate organizational					Quality System Control	Control Quality
1.1.3.1	Responsibility and Authority		Each manufacturer shall establish the appropriate responsibility, authority, and					Quality System Control	Control Quality
Level 2									
2.1.1	Applicable Regulations		The device product shall meet all applicable regulations and standards.		Product		Global Patient Safety	1. Demonstrate Compliance	1. Approved for Use
2.1.2	Usability		The device, including packaging and labeling, shall meet requirements for		Product		Global Patient Safety	1. Demonstrate Compliance	1. Approved for Use
2.2.1	Drug Product Quality during Manufacturing		The quality of the Drug Product shall be maintained throughout manufacturing of		Product		Manufacturing	2. Protect Product	2. Product Safety
Level 3									
3.2.1	Drug Product Quality during Manufacturing		The quality of the Drug Product shall be maintained throughout manufacturing of		Product		Manufacturing	2. Protect Product	2. Product Safety
3.2.2	Material Compatibility		unsafe reaction when exposed to the Drug Product Formulation.		System DIS	plastic materials do not react negatively	Manufacturing	2. Protect Product	2. Product Safety
3.2.3	Container Closure Pressurization		compromise the container and or needle sterility prior to use.		System DIS	manufacturing process does not	Manufacturing	2. Protect Product	2. Product Safety
3.2.4	Drug Product Quality during Distribution		The quality of the Drug Product shall be maintained within specification throughout distribution.		Product		Supply Chain	2. Protect Product	2. Product Safety
Level 4									
4.7.3	Exposed Needle Length		The insertion depth shall be 0.5 ± 1.0 mm from the base plate to the tip of the		System DIS	To maximize the probability of	Medical	7. Deliver Effective Dose	7. Maintain Health
4.7.5	Needle Gauge		The Delivery System container Closure shall use a 27 gauge thin wall or smaller		Design Guide	Marketing requirement.	Marketing	7. Deliver Effective Dose	7. Maintain Health

Figure 30: Part of Multi-level Requirement Model

## Implementation

### Excel

For implementation in Excel, we put the classified requirements in Excel sheet and embed code for functions to implement the model. Then multi-level requirements template for Excel comes out. This template can generate a new requirements model file just by choosing altered requirements in the database while updating. A demo of multi-level requirements template in Excel table is showing in Figure 31.



A	B	C	D	E	F	G	H	I	J	K	M	N
ID	Requirement/Spec Name	Functional Requirement	Requirement/Spec Description	Specific Value	Requirement/Spec Category	Rationale	Stakeholder	Functions	User Needs			
					To Level 2		To Level 3			Generate		
2	Level 1											
3	1.1.1	Quality System	Each manufacturer shall establish and maintain a quality system that is					Quality System Control	Control Quality			
4	1.1.2	Quality Policy	Management with executive responsibility shall establish its policy and objectives for,					Quality System Control	Control Quality			
5	1.1.3	Organization	Each manufacturer shall establish and maintain an adequate organizational					Quality System Control	Control Quality			
6	1.1.3.1	Responsibility and Authority	Each manufacturer shall establish the appropriate responsibility, authority, and					Quality system control	Control Quality			
78	Level 2					To Level 1 FDR		To Level 3				
79	2.1.1	Applicable Regulations	The device product shall meet all applicable regulations and standards.		Product		Global Patient Safety	1. Demonstrate Compliance	1. Approved for Use			
80	2.1.2	Usability	The device, including packaging and labeling, shall meet requirements for		Product		Global Patient Safety	1. Demonstrate Compliance	1. Approved for Use			
81	2.2.1	Drug Product Quality during Manufacturing	The quality of the Drug Product shall be maintained throughout manufacturing of		Product		Manufacturing	2. Protect Product	2. Product Safety			
82	2.2.2	Material Compatibility	The Device materials shall not have an unsafe reaction when exposed to the Drug		System DIS	To ensure the plastic materials do	Manufacturing	2. Protect Product	2. Product Safety			
152	Level 3					To Level 1 FDR		To Level 2				
153	3.2.1	Drug Product Quality during Manufacturing	The quality of the Drug Product shall be maintained throughout manufacturing of		Product		Manufacturing	2. Protect Product	2. Product Safety			
154	3.2.2	Material Compatibility	unsafe reaction when exposed to the Drug Product Formulation.		System DIS	plastic materials do not react negatively	Manufacturing	2. Protect Product	2. Product Safety			
155	3.2.3	Container Closure Pressurization	compromise the container and or needle sterility prior to use		System DIS	manufacturing process does not	Manufacturing	2. Protect Product	2. Product Safety			
156	3.2.4	Drug Product Quality during Distribution	The quality of the drug product shall be maintained within specification		Product		Supply Chain	2. Protect Product	2. Product Safety			
157	3.2.5	Shelf Life	The device shall have a shelf life in cold storage of not less than the shelf life		Product		Supply Chain	2. Protect Product	2. Product Safety			

Figure 31: Part of Template Table

Moreover, we embedded several functions in the excel file to implement this model. And Visual Basic for Application (VBA) is the default programming language of Excel. So we code in VBA programming these functions.

Figure 32 is a Requirements report, which contains all selected requirements in the requirements database, generated by the excel table. Our implementation tool can automatically generate a file containing the checked requirements. This function makes the Requirements Model accessible to varied management platform.

A	B	C	D	E	F	G	H	I	J	K	L
ID	Requirement/Spec Name	Functional Requirement	Requirement/Spec Description	Specific Value	Requirement/Spec Category	Rationale	Stakeholder	Functions	User Needs		
Requirements Report											
3	1.1.7	Quality Audit	Each manufacturer shall establish procedures for quality audits and cond.					Quality Sy	Control Quality		
4	1.1.8	Personnel	Each manufacturer shall have sufficient personnel with the necessary ed.					Quality Sy	Control Quality		
5	1.2.1	Design Control	Each manufacturer shall establish and maintain procedures to control the					Design Co	Control Design		
6	1.2.2	Design and Development Planni	Each manufacturer shall establish and maintain plans that describe or refc					Design Co	Control Design		
7	1.2.3	Design Input	Each manufacturer shall establish and maintain procedures to ensure that					Design Co	Control Design		
8	2.22.3.1	Storage Shelf Life	The shelf life of the Delivery System shc		System DIS	Device pe	Supply Ch	Protect Pr	Product Safety		
9	2.22.4	Anti-Counterfeiting a	The Product shall be designed with anti-		Product		Global Pat	Protect Pr	Product Safety		
10	2.22.4.1	Packaging Tamper Evi	GQS 302 forms will be submitted for the		System DIS	GQS 302	Global Pat	Protect Pr	Product Safety		
11	2.22.4.2	Anti -Counterfeiting	The Product shall provide anti-counterfe		System DIS	GQS 302	Global Pat	Protect Pr	Product Safety		
12	2.22.5	Drug Product Quality	The quality of the Drug Product shall be		Product	ISO 11608	Global Pat	Protect Pr	Product Safety		
13	2.23.1	Product Labeling	The Delivery System shall maintain the		System DIS	ISO 11608-5:2012 Sec	Protect Pr	Product Safety			
14	3.22.1	Drug Product Quality during Man	The quality of the Drug Product shall be		Product		Manufact	2. Protect	2. Product Safety		
15	3.22.1.1	Material Compatibility	The Device materials sh		System DIS	To ensure	Manufact	2. Protect	2. Product Safety		
16	3.22.1.2	Container Closure Pre	The NIS-Auto shall be d		System DIS	To ensure	Manufact	2. Protect	2. Product Safety		
17	3.22.2	Drug Product Quality during Dist	The quality of the Drug Product shall be		Product		Supply Ch	2. Protect	2. Product Safety		
18	3.22.3	Shelf Life	The device shall have a shelf life in cold		Product		Supply Ch	2. Protect	2. Product Safety		
19	3.22.5.1	Drug Product Quality Prior to Use	The Delivery System shall maintain the		Product	ISO 11608	Global Pat	2. Protect	2. Product Safety		
20	3.24.1.1	Weight	The Device with a filled container closur		System DIS	To be com	Marketing	4. Handle	4. Safe and Easy to Handle		
21	3.24.1.2	Body Upper Height	The Device body upper shall have a min		System DIS	MIL-STD-1	Marketing	4. Handle	4. Safe and Easy to Handle		

Figure 32: Part of Automatically Generated Requirement Report in Excel

The workflow of Excel implementation starts selecting the desired requirements under each level. And after finishing the selection of desired requirements, the final step is to generate a file which contains all the selected requirements.

As the workflow, the first coding function is linking checkbox with each individual requirement. In that way, the desired requirement can be selected just by clicking the corresponding checkbox. Of course, one of the most important functions is generating final selected requirements file. For the convenience, the template is also embedded in some other functions coding by VB. In the template, we programed functions that make it possible to check all child-items automatically. And the background color of the requirement will change once the item is selected.

Despite the requirements are tabular items, the relationships among requirements still need to store in the template. And that can remind users of these relationships while choosing requirements. One of the reminders shows in Figure 33 and 34. This reminder helps users to check the associated requirements automatically.

3.24.1.3	Maximum Effective Diameter	The Device (not including marking or label) shall have a nominal diameter of 0.13 in. (3.00 mm).	System DIS	market research on Nestor device	Marketing	4. Handle Product	4. Safe and Easy to Handle
3.24.1.4	Height	The overall nominal height of the device in capped form shall not exceed 0.13 in. (3.00 mm).	System DIS	market research on Nestor device	Marketing	4. Handle Product	4. Safe and Easy to Handle
3.24.1.5	Base Cap Diameter	The Device base cap shall have an effective diameter (maximum diameter) of 0.13 in. (3.00 mm).	System DIS	Per MIL-STD-1472G Figure 10b.	Marketing	4. Handle Product	4. Safe and Easy to Handle
3.24.1.6	Base Cap Height	The Device base cap shall have a minimum grip height of 0.13 in. (3.00 mm).	System DIS	Figure 8 Width of MIL-STD-1472G	Marketing	4. Handle Product	4. Safe and Easy to Handle
3.24.1.7	Upright Stability	The Device with a new container closure installed shall be stable when placed in its intended use position.	System DIS	Providing the capability of the device to be used	Global Patient Safety	4. Handle Product	4. Safe and Easy to Handle

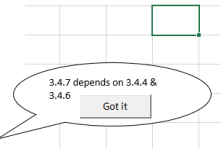


Figure 33: Relation Reminder

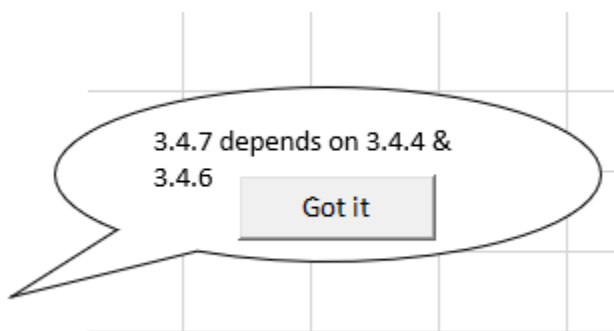


Figure 34: Partial Detail of Relation Reminder

The basic function is making every available requirement selectable by a checkbox. The user can click the checkbox to select corresponding requirements, which is needed in the practical project. All related requirements are saved in the template, so users can build the whole requirements file just by selecting the items they want. After saving the requirements in the template, we defined the relationships among requirements in the code. That comes to the second function. The template

can recognize the relations of each checked requirements, automatically selecting the dependent requirements, or blocking the conflicting requirements and showing the reminder at the same time. Also, there is an additional function for the child-parent relation. All sub-requirements are selected while the parent is checked, and vice versa. For the convenience, the code also makes the switching levels possible. The most important function is generating a new file. When users finish choosing the requirements what they want to develop their project, they can click the generate button to create a new file containing all the items they selected. And this file can be imported to their management platform.

### **Teamcenter Implementation**

For the implementation in an information system, we built a multi-level requirements template in Teamcenter in product structure management. In the template, we set the level of each requirement as one of their attributes, reference designator. The certain level of requirements can be selected in the template by groups, filtering the corresponding attributes. After defining the changed level of requirements, engineers can modify the entire level of requirements directly. The following Figure 35 shows that certain levels of requirements are selected in development structure in this template.

1.18.1/1;1-specified requirement servicing	Part	Working( )	10	None	e...	1
1.18.2/1;1-Analyzing servicing reports	Part	Working( )	20	None	e...	1
1.18.3/1;1-Report to FDA	Part	Working( )	30	None	e...	1
1.18.4/1;1-Documenting Service Reports	Part	Working( )	40	None	e...	1
1.19/1;1-Statistics	Part	Working( )	190	None	e...	1
1.19.1/1;1-Statistical techniques	Part	Working( )	10	None	e...	1
2.21/2;1-Approved for Use	Part	Working( )	200	None	e...	2
2.21.1/2;1-Applicable Regulations	Part	Working( )	10	None	e...	2
2.21.2/2;1-Usability	Part	Working( )	20	None	e...	2
2.22/2;1-Product Safety	Part	Working( )	210	None	e...	2
2.22.1/2;1-Drug Product Quality during Manufact...	Part	Working( )	10	None	e...	2
2.22.1.1/2;1-Material Compatibility	Part	Working( )	10	None	e...	2
2.22.1.2/2;1-Container Closure Pressurization	Part	Working( )	20	None	e...	2
2.22.2/2;1-Drug Product Quality during Distribution	Part	Working( )	20	None	e...	2
2.22.2.1/2;1-Packaging Protection	Part	Working( )	10	None	e...	2
2.22.2.2/2;1-Package Opening Requirements	Part	Working( )	20	None	e...	2
2.22.3/2;1-Shelf Life	Part	Working( )	30	None	e...	2
2.22.3.1/2;1-Storage Shelf Life	Part	Working( )	10	None	e...	2
2.22.4/2;1-Anti-Counterfeiting and Tamper Evide...	Part	Working( )	40	None	e...	2
2.22.4.1/2;1-Packaging Tamper Evidence	Part	Working( )	10	None	e...	2
2.22.4.2/2;1-Anti -Counterfeiting	Part	Working( )	20	None	e...	2
2.22.5/2;1-Drug Product Quality Prior to Use	Part	Working( )	50	None	e...	2
3.22.5.1/3;1-Drug Product Quality Prior to Use	Part	Working( )	10	None	e...	3
2.23/2;1-Instructions for Use	Part	Working( )	220	None	e...	2
2.23.1/2;1-Product Labeling	Part	Working( )	10	None	e...	2
2.23.1.1/2;2-Company Gold Requirements	Part	Working( )	10	None	e...	2
2.23.1.2/2;1-Non-language-Specific Labeling	Part	Working( )	20	None	e...	2
2.23.2/2;1-Industrial Design	Part	Working( )	20	None	e...	2
2.23.2.1/2;1-Product Differentiation	Part	Working( )	10	None	e...	2
2.24/2;1-Safe and Easy to Handle	Part	Working( )	230	None	e...	2
2.24.1/2;1-Form Factor	Part	Working( )	10	None	e...	2
3.24.1.1/3;1-Weight	Part	Working( )	10	None	e...	3
3.24.1.2/2;1-Body Upper Height	Part	Working( )	20	None	e...	3
3.24.1.3/2;1-Maximum Effective Diameter	Part	Working( )	30	None	e...	3
3.24.1.3/2;1-Maximum Effective Diameter	Part	Working( )	40	None	e...	3
2.24.2/2;1-Forces and Torques	Part	Working( )	20	None	e...	2
2.24.3/2;1-Safe Handling	Part	Working( )	30	None	e...	2
2.24.3.1/2;1-Sharp Edges, Pinch Points	Part	Working( )	10	None	e...	2
2.24.3.2/2;1-Biocompatibility	Part	Working( )	20	None	e...	2
2.24.3.3/2;1-Unlocked Indication	Part	Working( )	30	None	e...	2
2.24.3.5/2;1-Choking Prevention	Part	Working( )	40	None	e...	2
2.24.4/2;1-ISO Handling Requirements	Part	Working( )	40	None	e...	2

Figure 35: Search Level 3 in Template

After reviewing and updating requirement model copied from the template, a report can be generated automatically, showing in figure 36.

Requirements Report		
Date: 2018-05-31		
Project Id	Project Name	
0	Multiple Levels Req. Model	
Requirement Id	Requirement Name	Description
1.1	Control Quality	
1.1.1	Quality System	Each manufacturer shall establish and maintain a quality system that is appropriate for the specific medical device(s) designed or manufactured.
1.1.2	Quality Policy	Management with executive responsibility shall establish its policy and objectives for, and commitment to, quality.
1.1.3	Organization	Each manufacturer shall establish and maintain an adequate organizational structure to ensure that devices are designed and produced in accordance with the requirements of this part.
1.1.3.1	Responsibility and Authority	Each manufacturer shall establish the appropriate responsibility, authority, and interrelation of all personnel who manage, perform, and assess work affecting quality, and provide the independence and authority to perform tasks.
1.1.3.2	Resource	Each manufacturer shall provide adequate resources, including the assignment of trained personnel, for management, performance of work, and assessment activities, including internal quality audits, to meet the requirements of this part.
1.1.3.3	Management Representative	Management with executive responsibility shall appoint, and document such appointment of, a member of management who, irrespective of other responsibilities, shall have established authority over and responsibility.
2.21	Approved for Use	
2.21.1	Applicable Regulations	The device product shall meet all applicable regulations and standards.
2.21.2	Usability	The device, including packaging and labeling, shall meet requirements for usability per IEC 62366
2.22.2	Anti-Counterfeiting	The Product shall provide anti-counterfeiting meeting GQS 302
2.22.5	Drug Product Quality Prior to Use	The quality of the Drug Product shall be maintained until the dose is delivered in the patient.
3.22.5.1	Drug Product Quality Prior to Use	The quality of the Drug Product shall be maintained within specification throughout distribution.
2.23	Instructions for Use	
2.23.1	Product Labeling	The Product labeling shall comply with labeling Requirements established by the Company Labeling Department.
2.23.1.1	Company Gold Requirements	The Delivery System Labeling shall comply with labeling requirements established by the company Labeling Department.
2.23.1.2	Non-language-Specific Labeling	The embedded markings on the Device shall not be language specific and shall comply with ISO/FDIS 15223-2:2010
2.23.2	Industrial Design	The appearance of the Product shall be acceptable to the Target User and provide appropriate differentiation from other therapeutics found in autoinjectors.
2.23.2.1	Product Differentiation	The Product shall be designed to allow differentiation by dose strength and other similar products.
2.24	Safe and Easy to Handle	
2.24.1	Form Factor	The Device shall have an acceptable form-factor that allows the targeted user to use, transport, store, and dispose of it without the need for special equipment or methods.
3.24.1.1	Weight	The Device with a fitted container closure installed shall weigh no more than 60 grams.
3.24.1.2	Body Upper Height	The Device body upper shall have a minimum grip height of 0.5 in. (13.0 mm)
3.24.1.3	Maximum Effective Diameter	The Device (not including marking or label) shall have a nominal diameter (measured at top of Body Lower) of no greater than 22.5mm.
2.24.2	Forces and Torques	The forces and torques required to operate the Product shall enable safe and effective use.
2.24.3	Safe Handling	

Figure 36: Part of the Requirements Report Automatically Generated in Teamcenter

A requirement template was built in a tree form (similar to the BOM for product structure) in Teamcenter for future reuse. Figure 37 shows the workflow for using this template to develop a requirement model for a product. It's based on the basic workflow in figure 18 with several extra steps (showing in ovals). These are steps after defining the change of product design, from copy the template to searching and removing the related levels of requirements. For a new field product, there is no difference, starting with the very beginning regulatory requirements. For a new category product, the process begins after defining the changed parts in Teamcenter. Copy the template and paste in a new project, and then review and update the lower levels 2&3 (category and configuration). That means the regulatory requirements stay in a new project. For a new configuration, the process starts with the configuration, keeping the regulatory requirements and product requirements. Copy and paste the template, then review and update the level 3 (configuration requirements). With the template in Teamcenter, after decided the change level of the project, engineers can just duplicate all requirements in the template, and then review and update the related levels of requirements in the template.

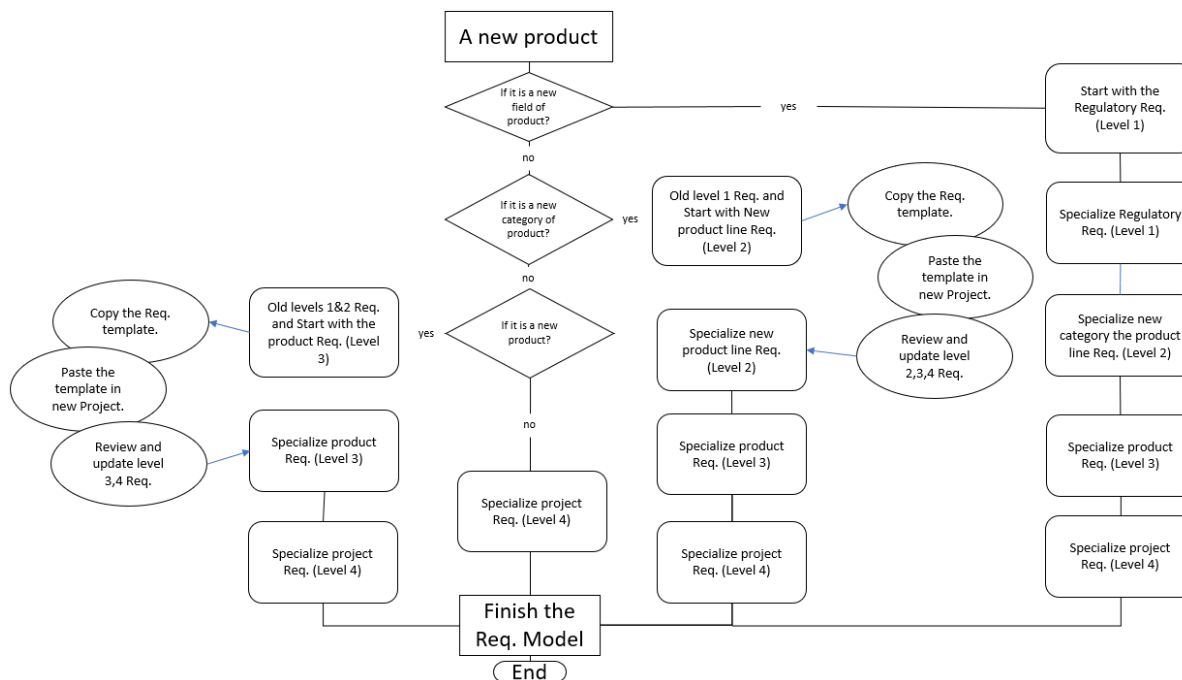


Figure 37: Multi-level Requirement Model Development Workflow in Teamcenter

With the template in Teamcenter, after defining the change level in a project, engineers can just review the template, and then update related levels of requirements in the template, which are the requirements needed to change in the new product. Based on the template, brand new requirements can be added following the development process, deriving from user need to requirements, and labeled their levels. If a new requirement is just new from an original user need, it only needs to add new requirements under the same user need.

Mainly there may be two types of change. For a specific project, there must be some changed requirements, including adding new requirements, deleting or modifying original requirements. Another is about the permanent change of the template, which refers to the maintenance.

For the first kind of change, generally, it happens for each project. The frequency differs from the levels. The regulatory requirements are most stable. They are only changed when the corresponding department changes the policy for this product field. The lower level's requirements change more frequently. For the second change, it happens when the product changes some important features or regulations. So, for the future convenience of reuse, the requirement template needs to make the corresponding change permanently.

The following figure shows the workflow of requirement change in Teamcenter template. While reusing this template in Teamcenter, if there is a requirement needed to change, the steps follow

like figure 38 The first step is defining that if the new requirement is derived from a new user need. If the answer is yes, the development starts with user need and determine their levels. After finishing the development process, create these items in Teamcenter template and label their levels. If the requirement is derived from original user need, then specify this requirement, create it under the affiliated user need and label it as the corresponding level in Teamcenter. If this requirement is not absolutely new, it is only needed to modify original details in the template. Besides, for the permanent change of the requirements template, the workflow is also same as these steps showing in figure 38.

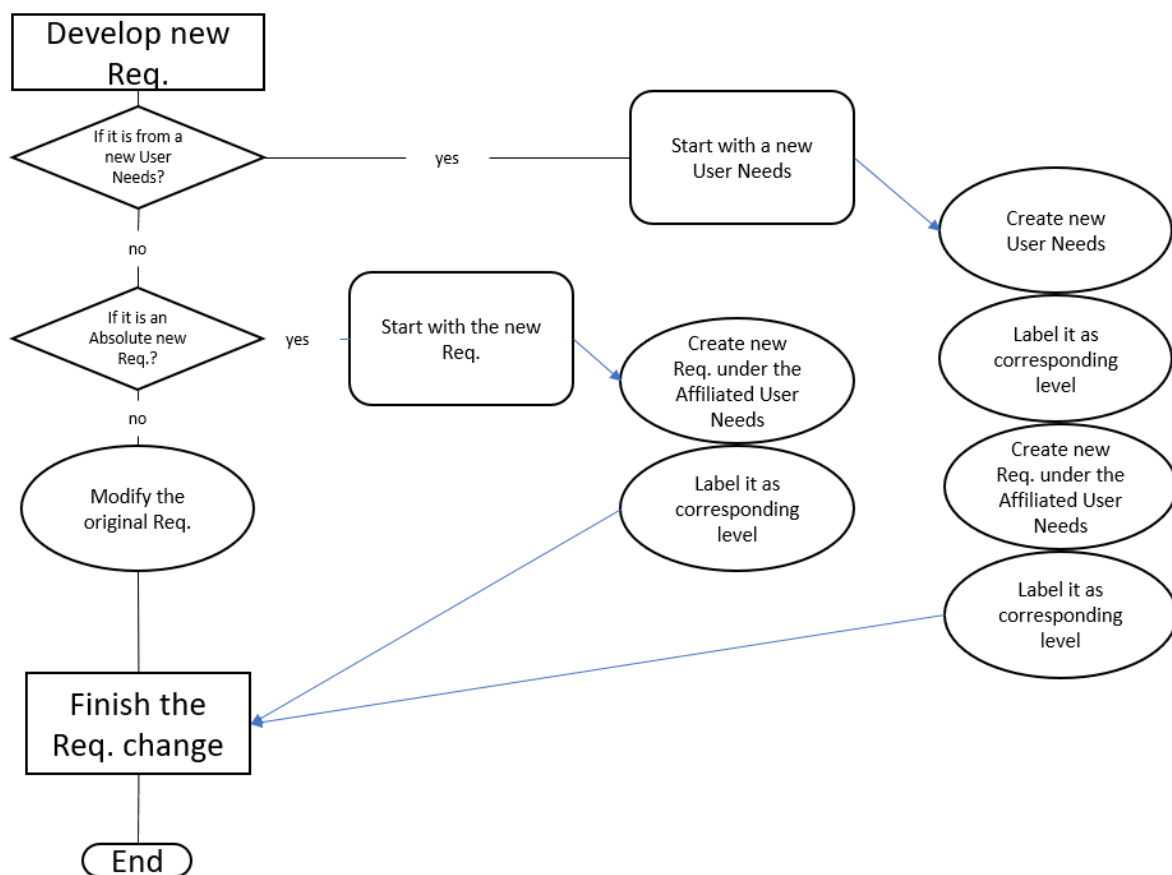


Figure 38: Requirement Change Workflow in Teamcenter

### Comparison between Two Implementations

We chose two different software tools, the MS Office Excel and Siemens Teamcenter, for implementation of the multi-level requirement model. There are some similarities between the two

implementations. First, both of them support automatic generation of requirement documents after the necessary revise of the template. Second, both of them group the requirements based on their levels for easy selections. However, there are also some differences between the two implementations. Excel is the most commonly used tool for requirements documentation, and it is also a tool that most of the companies have already invested in and most of the engineers are familiar with. The Excel implementation will be easier for companies and engineers to accept and adopt the new reuse approach. This implementation can be used as a standalone requirement management tool, or as a middleware or interface to be integrated with other software tools. Teamcenter, as the leading PLM platform, can provide more powerful support on the management of the requirements; and more importantly, it provides an opportunity to connect requirement data with other product development related data (BOM, materials, manufacturing process, etc.), since it can be the only data source for an entire enterprise. We want to use these two different types of tools and implementations to show the feasibility and flexibility of our multi-level requirement model.



## Benefits of the Multi-level Requirement Model and its Implementation

### Avoiding Potential Pitfalls

Table 3 showed how the new reuse approach with the multi-level requirement model can help to reduce the steps in generation new requirement documents and avoid the pitfalls mentioned in Table 1.

Table 3: Avoiding Potential Pitfalls with New Reuse Approach

Step #	Description of the Step	Potential Pitfalls	New Process
1	When an auto-injector is chosen as the delivery mechanism for a drug, the product engineer needs to find the requirement document of a similar product. Then copy the document for the new project.	<ul style="list-style-type: none"> <li>• Wrong product, the product may not be the one that is the closest related to the new project.</li> <li>• Right product, but wrong version of the document.</li> </ul>	<ul style="list-style-type: none"> <li>• The software tool will help to locate the right document for the right product.</li> <li>• The continuously review and maintenance process ensures that the up-to-date document will be used.</li> </ul>
2	Review the entire file to identify the reusable ones, for example, the regulatory related requirements, and the ones that need to be updated or removed, for example, requirements related to needle gauge and labeling.	<ul style="list-style-type: none"> <li>• Miss identifying reusable requirements</li> <li>• Longer time to go through the entire documents</li> </ul>	<ul style="list-style-type: none"> <li>• The templates are pre-defined and carefully reviewed by domain experts to make sure that the requirements are at the right level for reuse.</li> <li>• The software tools can provide easy query functions to save the time.</li> </ul>
3	Revise the non-reusable ones and add new ones as needed.	<ul style="list-style-type: none"> <li>• Revised or newly added requirements contradictory or duplicate to the existing ones</li> </ul>	<ul style="list-style-type: none"> <li>• Build-in compliance and completeness check will help to improve the quality of the requirements.</li> </ul>

Besides all the improvements mentioned in Table 3.1, both of the two implementations support automatic requirement document generation which can save time and avoid possible human errors in creating documents.

### Increasing the Number of Reusable Requirements

There are 184 requirements in total, with 75 in Level 1, 73 in Level 2, 29 in Level 3, and 7 in Level 4 (Table 4). For a product reconfiguration project (for example, a new drug using an existing auto-injector), 96% of the requirements can be reused. More than 40% of the requirements can be used for a product in a brand new product line, while about 80% requirements can be reused for a product in an existing product line (Figure 39).

Table 4: Statistics of Requirements Number

Statistics of Requirements Number	
Total Number of Requirements	184
Number of Requirements in Level 1	75
Number of Requirements in Level 2	73
Number of Requirements in Level 3	29
Number of Requirements in Level 4	7

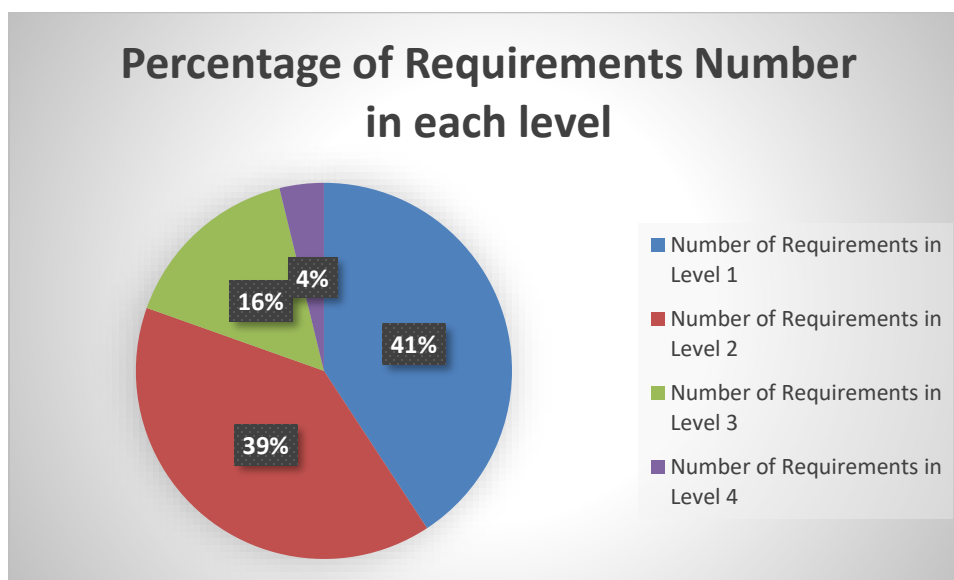


Figure 39: Percentage of Requirements Number in each Level

## DISCUSSIONS

In the past few decades, many companies start adopting the idea of product configuration and customization in order to keep up with the rapid changing technologies and satisfy the significant increasing on customized and personalized products from customers. Instead of developing new product from scratch each time, product configuration allow the companies to build their new product based on a group of pre-developed and validated parts or technologies. Reuse of existing designs and technologies becomes a widely adopted approach. As a key step in a product development process, requirement engineering is also embracing the reusable idea in requirement development and management. However, how to effectively reuse the existing requirement data remains an unanswered question. Some companies are trying to take advantage of software tools (either standalone or a part of a larger information platform) to improve the reusing of requirement data. But there is a clear lack of knowledge on the data model for requirement management and the workflow to create, maintain, search, and retrieve requirement data from the tool. So, the objectives of the current project are to (1) develop a requirement data management model to support easy reuse of the data; (2) develop corresponding workflows and guidelines on how the data model working; and (3) implement the model and the workflow into two different software tools; (4) assess the benefits of the multi-level requirement model. Following the research approach discussed in Chapter 2, we built the data management model and the workflow, and also successfully implemented them into two different software tools to verify the feasibility of the model and the workflow.

Our results presented in Chapter 3 showed that it is possible to implement the multi-level model in both Excel and Teamcenter. With the new process, the multi-level model could avoid the pitfalls of current reusing way mentioned in Chapter 1. Following the workflow in results, engineer could find the right document for reusing. And the maintenance ensures the reused document is up-to-date. While reusing, because the level of requirements in the templates derived in Chapter 3 are predefined by experts, reusing process could reduce the miss identifying of reusable requirements. Besides, the query functions in both implementations reduce the reviewing time. Also, it increases the reusable of the requirements. Most part of requirements could be reused in new project.

During the research, we had several observations beside the results shown in Chapter 3.

1. Managing requirements as documents in a software tool won't be able to have the full benefit from the tool. Companies should consider managing each requirement as a structuralized database item to make the searching, retrieving, changing, tracing and controlling of access of the requirement data easier. When a requirement report is needed, most of the software tools have the capability to automatically generate the report documents.
2. The requirement templates built in the software tool are not only a critical part of the requirements reusing process, but also a very helpful way to ensure that the requirement knowledge will remain inside the company and will become a useful tool to train the new generation of engineers. The requirement data and requirement template building process will need contributions from many different domain experts. These inputs from the engineers many years of experiences within their area are valuable assets to the company, and the template is a convenient and effective way to capture and management them. These templates also provide a good starting point for a new product/requirement/system engineer to begin their work. They can develop their new requirements/requirement models based on a validated foundation.
3. A well-defined workflow is very important to the success implementation of the model. A lot of concerns with requirement engineering/modeling are related to the usability of the models, during our interviews with people from industry. Most people agree that requirements are important and requirement management is necessary. But they are worried that more works will be introduced and they will their control and flexibility over their own works during the process. A clearly defined workflow together with guidelines on how to use and maintain the model will help people to accept the implementation, and also avoid confusions in the future.

Overall, the model and workflow developed in the study provide a more systematical approach to manage requirement data, especially with a software tool. Following are some of the future works which can help to generalize the model for a larger range of application and expand the scope of the model to cover more processes within product development.

1. Due to the time constrain, the model and workflow didn't get implemented in a real world working environment. A real working environment will bring in a lot more

uncertainties and variations. An implementation in such an environment will help to improve the current model and workflow to make it a more robust solution.

2. The current application is for one product line in medical devices industry—the auto injector. The application should be expanded both horizontally and vertically in the future. Horizontally, the application should cover a broad range of medical devices and even other industries; while vertically, the other product development related data should be included, for example, the testing/validation methods and results, the product structure, etc.

## CONCLUSIONS

Following conclusions were drawn based on the results from this study:

- A practical multi-level requirements management model for a medical device—the auto injector can be developed, and the model can be implemented into different software tools to support reuse of existing requirement data in creating requirement models for new product development projects.
- The workflow and guideline to support the application and maintenance of the requirement model can be developed and implemented. Requirement documents/reports can be automatically generated through the software tool by following the workflow.
- The new method can improve the reusability of requirement model. With the help of the multi-level model, it can avoid the pitfalls of current way in reusing requirements, increase the reusable requirements.

## REFERENCES

- Philippe Massonet, Alvaro E. Arenas, Christophe Ponsard, Benjamin Aziz. (2015). *Goal-Oriented Requirement Engineering Support for Business Continuity Planning*. Paper presented at the International Conference on Conceptual Modeling. Lecture Notes in Computer Science retrieved from
- Bernhard Rumpe, Arvid Butting, Christoph Schulze, Ulrike Thomas, Andreas Wortmann. (2016). Modeling Reusable, Platform-Independent Robot Assembly Processes. *arXiv*, 1061.02452v1.
- Claes Wohlin, Aybüke Aurum. (2005). *Engineering and Managing Software Requirements*: Springer.
- Laurent Balmelli. (2007). An Overview of the System Modeling Language for Products and Systems Development. *Journal of Object Technology*, 6, 149-177.
- Nicolas Guelfi, Barbara Gallina. (2007). A Template for Requirement Elicitation of Dependable Product Lines. *Springer*, 63-77.
- Barry Boehm. (2000). Spiral Development: Experience, Principles, and Refinements. *Special Report CMU/SEI-2000-SR-008*.
- Michael Waite, Chanaka Aluwihare, Christopher French. (2014). *MBSE: A Methodology for Collaborative Requirements Engineering*. Paper presented at the Australian Defence Engineering Conference 2014
- Robert Darimont, Christophe Ponsard, Arnaud Michot. (2015). *Combining Models, Diagrams and Tables for Efficient Requirements Engineering- Lessons Learned from the Industry*. Paper presented at the INFORSID 2015, Biarritz.
- David Powers, Davoud Mougouei. (2016). GOTM a Goal-oriented Framework for Capturing Uncertainty of Medical Treatments. *arXiv*, abs/1612.02904.
- Chris Doig. (2015, October 8). Why capturing enterprise software requirements is so difficult. Retrieved from <https://www.cio.com/article/2990512/enterprise-software/why-capturing-enterprise-software-requirements-is-so-difficult.html>
- Achim Rettberg, Fabiola Goncalves C. Ribeiro, Carlos E. Pereira, Michel S. Soares. (2017). *A Model-Based Engineering Methodology for Requirements and Formal Design of Embedded and Real-Time Systems*. Paper presented at the the 50th Hawaii international Conference on System Sciences, Hawaii.
- Cesare Fantuzzia, Giacomo Barbieria, Roberto Borsarib. (2014). A model-based design methodology for the development of mechatronic. *Mechatronics*, 24(7), 833-843. doi:0.1016/j.mechatronics.2013.12.004
- Gunnar Hedlund. (1994). A Model of Knowledge Mangement and the N-form Corporation. *Strategic Management Journal*, 15, 73-90.
- IEEE. (1998). 1220-1998 - IEEE Standard for Application and Management of the Systems Engineering Process. from IEEE
- Fatma Basak Aydemir, Jennifer Horkoff, Evellin Cardoso, Tong Li, Alejandro Mate, Elda Paja, Mattia Salnitri, John Mylopoulos, Paolo Giorgini. (2016). Goal-Oriented Requirements Engineering: A Systematic Literature Map. 106-115. doi:10.1109/re.2016.41
- Ken Jackson, Jeremy Dick. Elizabeth Hull. (2017). *Requirements Engineering* (4th ed.): Springer.
- JoAnn Giar, John Girard. (2015). Defining knowledge management: Toward an applied compendium. *Online Journal of Applied Knowledge Management*, 3(1), 1-20.

- Hal Mooz Kevin Forsberg, Howard Cotterman. (2005). *Visualizing Project Management: Models and Frameworks for Mastering Complex System* (3rd ed.). new york: Wiley.
- Begona Lloria. (2008). A review of the main approaches to knowledge management. *Knowledge Management Research & Practice*, 6(1), 77-89.
- Dorothy E. Leidner, Maryam Alavi. (2001). Knowledge Management and Knowledge Management System: Conceptual Foundations and Research Issues. *MIS Quarterly*, 25, 107-136.
- Jos Vrancken, Michel dos Santos Soares. (2007). *Requirements Specification and Modeling through SysML*. Paper presented at the 2007 IEEE International Conference on Systems, Man and Cybernetics, Montreal, Que., Canada.
- MITRE. (2015, August 20). Eliciting, Collecting, and Developing Requirements. Retrieved from <https://www.mitre.org/publications/systems-engineering-guide/se-lifecycle-building-blocks/requirements-engineering/eliciting-collecting-and-developing-requirements>
- Oracle (Producer). (2018, 2018). AGILE CUSTOMER NEEDS MANAGEMENT. *ORACLE DATA SHEET*. Retrieved from <http://www.oracle.com/us/products/applications/agile/customer-needs-management-ds-081122.pdf>
- Wei Zhao, Robert Darimont, Christophe Ponsard, Arnaud Michot. (2016). *A Modular Requirements Engineering Framework for Web-based Toolchain Integration*. Paper presented at the IEEE 24th International Requirements Engineering Conference, Beijing, China.
- Mehrdad Sabetzadeh, Shiva Nejati, Chetan Arora, Lionel C. Briand, Felix Mandoux. (2016). *Automated change impact analysis between SysML models of requirements and design*. Paper presented at the the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, Seattle.
- Siemens. Requirements Management. Retrieved from <https://www.plm.automation.siemens.com/en/products/teamcenter/requirements-management/>
- Standish. (2014). *Chaos report*. Retrieved from <https://www.projectsmart.co.uk/white-papers/chaos-report.pdf>
- Karen Willcox, Tan Bui-Thanh, Omar Ghattas, Bart van Bloemen Waanders. (2007). Goal-oriented, model-constrained optimization for reduction of large-scale systems. *Journal of Computational Physics*, 224(2), 880-896. doi:10.1016/j.jcp.2006.10.026
- Wikipedia. (2018, February 10). Product Structure. Retrieved from [https://en.wikipedia.org/wiki/Product\\_structure\\_modeling](https://en.wikipedia.org/wiki/Product_structure_modeling)
- Laurie Williams. (2004). Use Case-based Requirements. *SE Materials*, 2-5.