MASTERS THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF **MASTER OF SCIENCE**

TITLE: Vehicle Feature Complexity Modeling and Management in the SysML

PRESENTED BY: Jesus Mata Castañeda, Natalie Matevossyan

ACCEPTED BY:

_____

Advisor, *Michael Vinarcik* **Date**

_____

Department Chairperson, *Dr. Darrell Kleinke* **Date**

APPROVAL:

_____

Dean, *Gary Kuleck* **Date**
College of Engineering and Science

MASTERS THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF **MASTER OF SCIENCE**

TITLE: Vehicle Feature Complexity Modeling and Management in SysML

PRESENTED BY: Jesus Mata Castañeda, Natalie Matevossyan

ACCEPTED BY:

_____

Committee Member, *Dr. Jonathan Weaver, Ph.D.* **Date**

_____

Committee Member, *Dr. Darrell Kleinke, Ph.D.* **Date**

Dedication

Jesus Mata Castañeda

This thesis work is dedicated to my wife Diana, who has been a constant source of motivation and inspiration during the challenges of this program. I love her and am very thankful to have her in my life. I also want to dedicate this work to my parents Lucy and Gustavo that have always being a role model for me and have supported and encouraged all my endeavors.

Natalie Matevossyan

I am thankful for the patience and encouragement my Mom always provided me with, and for her never having any doubts that I will succeed in anything I do. I am thankful for two bright lights in my life my daughter Angelina and my son Alexander. I lovingly dedicate this work to them.

Abstract

Vehicle complexity management is one of the most challenging topics in the Automotive Industry. In the past, vehicle manufacturers offered the market unique models with the small variety of options. Nowadays, with all the emerging technology and consumer-oriented markets, vehicle feature packages become more and more complex. The complexity management has become a very challenging task for the OEMs' manufacturing.

This thesis will be focused on one of the most used documents in the industry which we will call a **Vehicle Feature Code Matrix** (VFCM) for the generalization purposes. We introduce a "Proof of Concept" and explore the possibilities of creating a dynamic SysML model for this document by applying the Zero Defect VFCM practices and generic Rules for VFCM creation in SysML.

Acknowledgements

We would like to thank our Adviser, Professor Michael Vinarcik for his extensive guidance and support as we worked on the thesis.

We would like also to recognize our immediate supervisors, especially, John Pawloski, and our teammates for their continuous support during our involvement in this program.

We also wish to recognize our families for understanding, support and encouragement that we needed during this program.

Contents

a. List of Figures

b. List of Tables

Introduction

As the technology rapidly evolves in the 21st century, and customers articulate their increasing needs and wants more and more in today's consumer market, the feature complexity for the vehicle is becoming more and more challenging for the OEMs. The central strategic question with regards to the product diversity concerns the 'optimum' or 'appropriate' level of variety: on one hand, offering variety increases cost, on the other hand, it can provide product differentiation in the market, thus leading to higher market share and sales volume (H. Schleich, 2007). Hence, it is imperative to find the balance between the complexity management and cost reduction. Some of the tension can be lessened by using the Vehicle Feature Code Matrix which is the authoritative document that controls and communicates approved content and any content changes for vehicle programs from Program Start to Job Last.

The goal of this thesis is to develop a "Proof of Concept" and present a different approach to creating, maintaining and managing the VFCM using SysML parametric model. We will discuss the inputs and outputs of the VFCM, discuss the "Proof of Concept" for the VFCM modeled in SysML to help to standardize and automate the VFCM management. We will discuss developing some special reports to support specific engineering activities to allow more flexibility and the VFCM usage efficiency (e.g., Cost Studies)

Chapter 1 discusses the vehicle complexity problem in general, contents and structure of the VFCM document and its current state, potential issues in the legacy process and the opportunities that were identified throughout the course of our research.

Chapter 2 discusses the Concept Design steps we went through in more details.

Chapter 3 describes the SysML Modeling general principles and terminology.

Chapter 4 discusses Systems Thinking for the VFCM Modeling and contains more detailed information on the usage of the SysML approach for creating a parametric model of the VFCM.

Chapter 5 discusses the new VFCM Functionality and the cost rollup analysis as a part of the Proof of Concept

Chapter 6 discusses potential next steps and further research opportunities.

Chapter 7 contains the conclusions and recommendation for further development of the VFCM and related model and documentation.

1. Vehicle Feature Complexity

Vehicle Program is considered a highly complex technical project and the VFCM is the one of the most important documents to manage the program feature complexity. The traditional VFCM is a “Document” that is intended to capture vehicle program changes and maintain assumptions of content deployment. Given the dynamic nature of the vehicle product development, the program assumptions constantly change and are adapted based on the inputs of various stakeholders.

The VFCM is the primary source of direction for a vehicle line; it documents all the complexity of options that is able to be built (engineered), and what is going to be built for specific markets or market grouping. It defines the timing points, vehicle series, standard and optional content, plants, markets, etc. It delivers the required codes for Marketing, Finance, Engineering, etc.

The VFCM in principle is a very active document that continually collects inputs from Engineering and Marketing, and evolves with new information generated during the program development. The changes in the VFCM have a very significant impact in the vehicle development, and are the official source where the related departments get information to do their own work to contribute to the development of the vehicle.

Current Format of the VFCM

The current execution of the VFCM is a very good example of a "document-based" approach to engineering, as described by Delligatti (Delligatti, 2013, p. 2) "***the document-based approach to systems engineering is expensive***"; this approach requires to manually update the document, generate and maintain multiple versions, generate disjoint documents, cascade updates and make sure that all the derivative documents or clients who use the VFCM are up to date with the latest changes.

The current format of the VFCM is a static document published in .pdf or Microsoft Excel format; when new information is available due to a vehicle feature change or update, a new version of the document is published and replaces the previous version.

Structure and Contents of the VFCM

The VFCM as it is available to the users today comes in the MS Excel form with various tabs which contain overlapping information and are updated and managed manually. Common main tabs of the VFCM consist of but not limited to the following:

- Summary
- Market List
- Legal and Mandatory Engineering Features by Market
- Availability and Deployment of the Power packs by Market
- Features and Options

- Deployed Features by Market
- Packages
- Navigational Data, etc.

The information contained in the various tabs is repeated in different formats which opens up an opportunity for a more effective approach where all the information can be linked together in a more reliable and user-error proof way.

Terminology Used

S Standard Feature (This is the Default feature choice; however, it can be replaced by an Optional feature in Marketing)

O Optional Feature (This is an engineered feature that can replace a Standard feature)

C Contains (This is used to define packages and assign a feature as the package owner)

I Includes (This is used to define package features owned elements, and all Included features have an owner that contains them)

M Mandatory (This is a feature strongly needed by the customer, or required for technical engineering reasons)

L Legally Required Feature (This is a feature required by law in the markets where the vehicle is sold)

Potential Issues in the Legacy Process of the VFCM

Since the VFCM contains a large volume of information the document may grows over 100 pages; and the way it is presented causes difficulties in finding information, understanding the directions, and keeping all the engineering team timely informed on the last program direction. It does not allow for parametric changes and requires a lot of manual rework to maintain the document up-to-date. Errors in the VFCM are detected only after they were released in a new version of the VFCM and the users start interacting with the data. It also allows for a user-error if not followed accordingly. This is especially costly for the late changes.

It is very common to release new versions of the VFCM a few times during the Vehicle Builds and have to dedicate a significant amount of time and resources to error-proof the VFCM so it is consistent internally and has no contradictions; the text in the document can become complex when capturing change over change. Additionally, every text, matrix or presentation that uses VFCM as the source of data has to be manually updated to reflect the latest changes.

NOTE: The key potential issue in the legacy process: the VFCM is 100% manually authored, generated, and managed.

2. Concept Design

We will consider the VFCM to be the "Product" and as with every product development there are stages that design must go through.

The systems engineering processes begin very simply with the identification of a need for a new or improved system (R. Ian Faulconbridge, 2003)

Conceptual design is normally the domain of the customer who is responsible for determining the system needs. Four major tasks associated with the conceptual design are described below (see Figure 2-1).

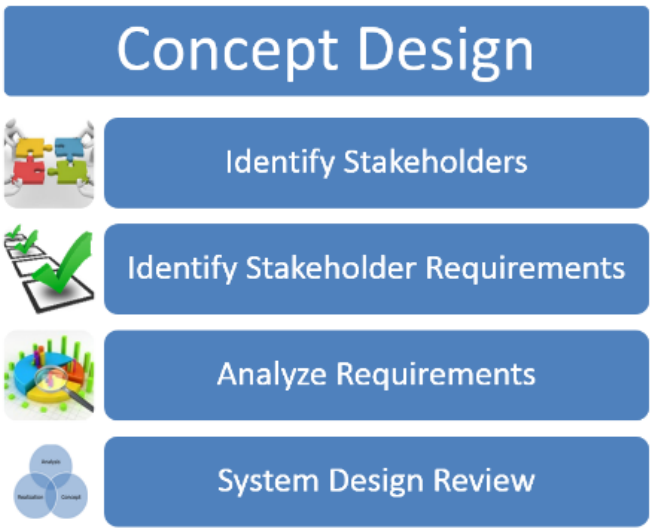


Figure 2-1 Concept Design Stages

Identifying Stakeholders

Before any work can commence on developing the system, the basic stakeholder requirements must be clearly and completely articulated (Faulconbridge, 2002).

Stakeholders for this project are identified below:

- VFCM Authors
- Program Teams
- Marketing Team
- Purchasing Team
- Finance Team
- Dealers

VFCM Author is a person responsible for inputting and verifying all necessary information in the VFCM **Program Team** is a collective user of the VFCM information

Marketing Team is the team which is a heavy user of the VFCM information in order to meet the requirements of different markets and markets grouping

Purchasing Team uses VFCM data to make the decision about purchasing particular components for specific Feature Codes for different variants

Finance Team is a very interested potential stakeholder which will be able to use the model for cost roll-up and cost analysis purposes

Dealers are not necessarily affected directly by VFCM changes but still use information for vehicle configuration determination and serve as an interface and the messenger of the available features to the customer

Identifying Stakeholder Requirements

Identifying Stakeholder Requirement Table is shown below on Figure 2-2. The initial constraints needed for the proof of concept have been identified on the project, and the design levels.

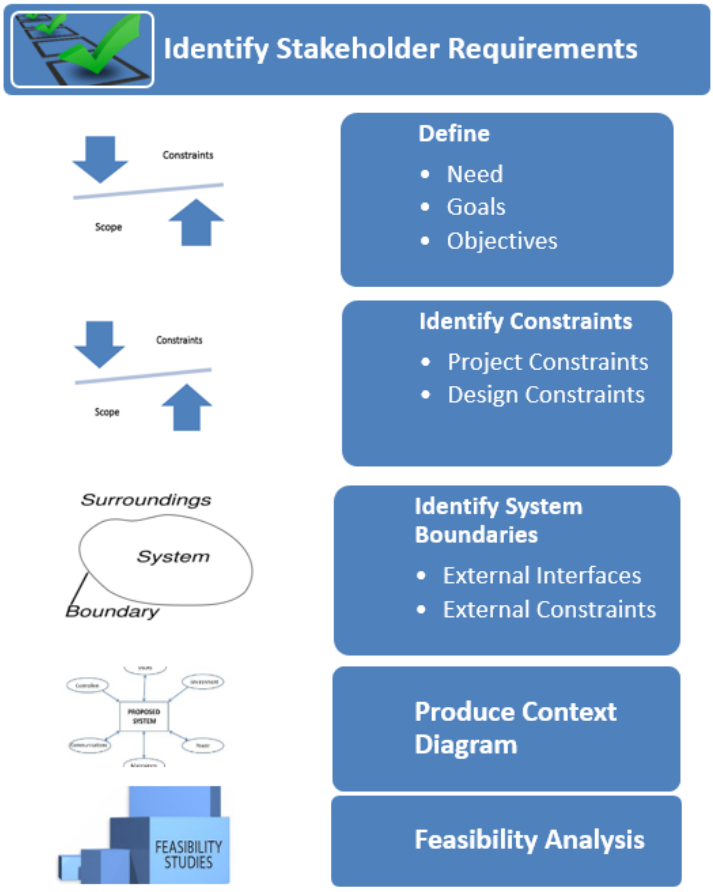


Figure 2-2 Steps to Identify Stakeholder Requirements

In order to identify Stakeholder Requirements Needs, Goals, and Objectives must be defined as shown on Figure 2-3.

Figure 2-3 Stakeholders Needs, Goals and Objectives

The preliminary requirements to satisfy Stakeholders' needs are shown on the Figure 2-4 Kano Model for Preliminary RequirementsFigure 2-4

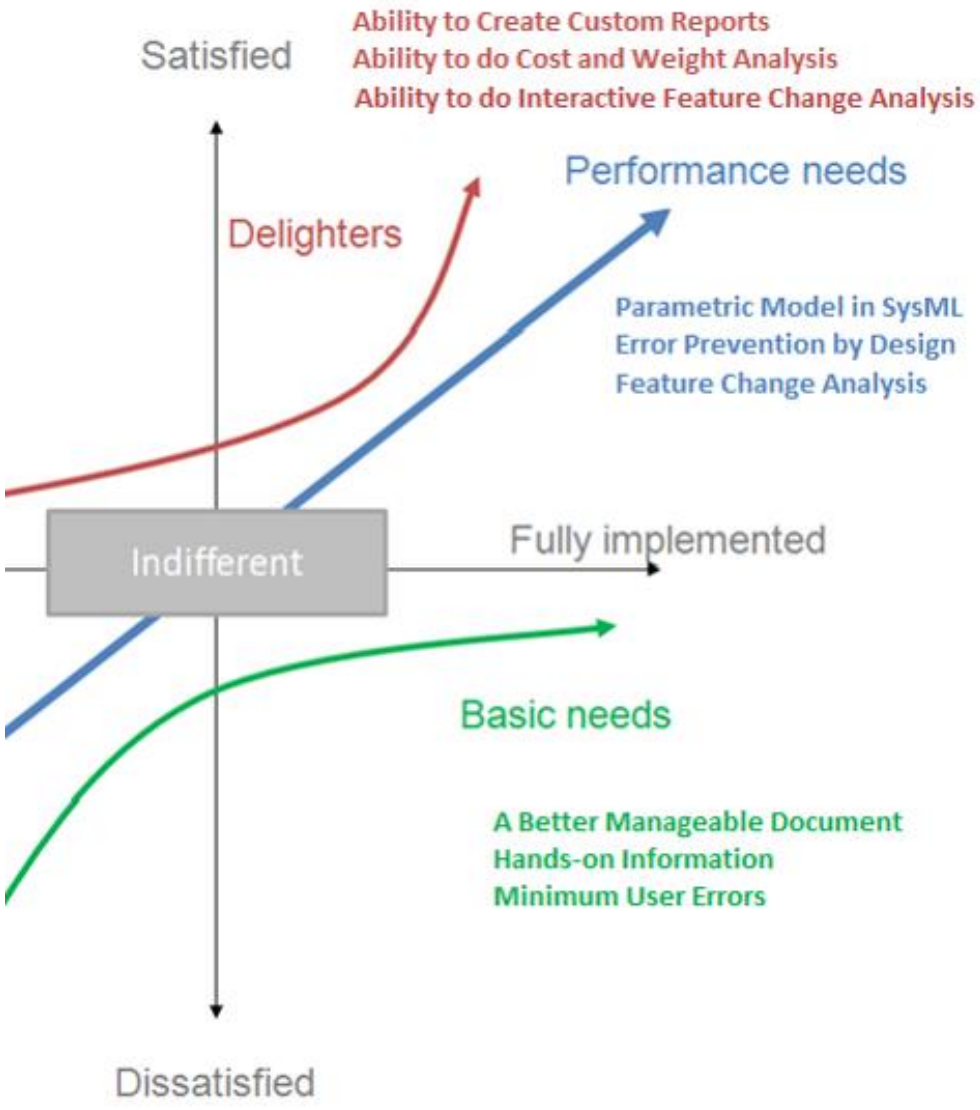


Figure 2-4 Kano Model for Preliminary Requirements

Basic Needs or Threshold (or basic) attributes are the expected attributes or "musts" of a product, and do not provide an opportunity for product differentiation (Deployment, 2005). In this case a better manageable document

with hands-on information available and minimum user-errors expected are the basic needs.

Performance attributes are those for which more is generally better, and will improve customer satisfaction (Deployment, 2005). In this case, Error Prevention by Design and Feature Change Analysis along with the parametric and not "document-based" model are considered to be the Performance Attributes.

Excitement Attributes or Delighters are unexpected features that will rise the customer satisfaction. Excitement attributes often satisfy latent needs – real needs of which customers are currently unaware. In this case, providing different custom reports, ability to do Weight and Cost studies, and an Interactive Feature Change Analysis are considered the Delighters.

Identify Project Constraints

The following project constraints were encountered during the research:

- Unavailability of the source raw data
- Working with highly confidential information

Identify External Constraints

One of the external constraints that will need to be taken into account is the VFCM Output data compatibility as an input into other systems.

External Interfaces

Interface control consists of establishing common understanding of interfaces for all project participants. (Weiss, 2013). The following diagram shows a SysML Context Diagram with the relations and interfaces of the VFCM as well as the related stakeholders and boundary systems (see Figure 2-5)

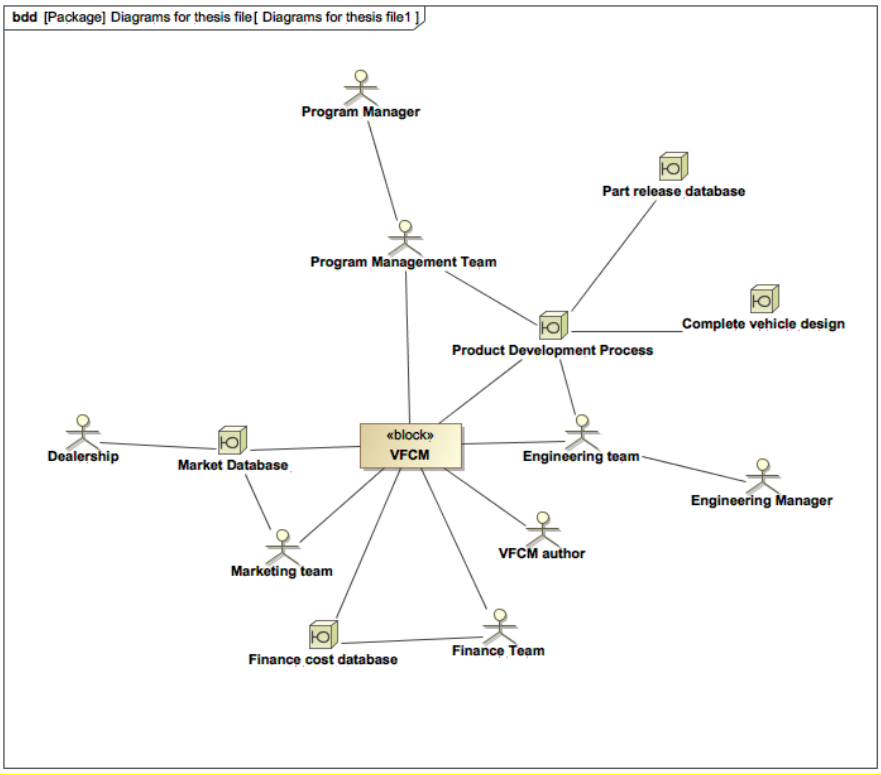


Figure 2-5: VFCM System Context Diagram

3. *SysML Modeling General Principles and Terminology Overview*

As opposed to the "document-based" approach there is a "model-based" System Engineering approach which proposes to generate an "integrated, coherent, and consistent System Model, created by using a systems model tool"

(Delligatti, 2013, p. 3). With the System Model the traditional document text information can be represented as model elements with a set of relationships between them. It also allows to re-use the model elements in diverse diagrams, tables, matrices as pure representation; the same model element can appear with different display methods but its characteristics are always referred to the original model element. The specification and characteristics of every model element are consistent throughout the entire model and changes to the elements propagate throughout the model to every element representation. This has the potential to significantly reduce the cost, time and effort to verify the consistency of documents, this is done almost automatically with support from a modeling tool.

SysML Modeling

According to the Collins dictionary "A model of a system or process is a theoretical description that can help you understand how the system or process works, or how it might work" (Collins-Dictionary, 2016). When a model is created, a language is being spoken (Delligatti, 2013, p. 5), not the traditional language but a modified language to set conventions and allow different people to understand the reasoning behind the representation of the reality.

Modeling has been used for long time by engineers to represent systems and conduct analysis. In the mid-1990s, a common modeling language called **UML** (Unified Modeling Language) was developed by Booch, Rumbaugh, and Jacobson. This language had a predominant focus on software development and have been adopted as a standard by the software community and is widely used throughout industry and government (Kossiakof, Sweet, Seymour, & Biemer,

2011). Since then UML has been a reference and has been updated and has increased popularity, the Systems Engineering community adapted to this language acclimatizing to some of the physiological barrier of software-oriented language (Kordon, Hugues, Canals, & Dohet, 2013).

In a certain way, SysML language is inspired in UML v2 as initially was released on 2007; however, "it includes the possibility of representing systems requirements, non-software components (mechanics, hydraulics, sensors, etc.), physical equations, continuous flows and allocations" (Kordon, Hugues, Canals, & Dohet, 2013, p. 48).

SysML is a modeling language that can be understood as a graphical language. "Its vocabulary consists of graphical notations that have specific meanings from an arrow to a solid line. One of the purposes of the language is visualization and communication of a system's design among stakeholders" (Delligatti, 2013, p. 12). The standardization of the language is managed by the OMG (Object Management Group). Similarly, to SysML there are other graphical modeling languages like the previously mentioned UML, UPDM, BPMN, MARTE, SoAML, IDEFx and other text languages like Verilog and Modelica (Delligatti, 2013, p. 5)

SysML modeling is based on the graphical representation of the system; and in order to standardize the visualization there are nine types of diagrams (refer to Figure 3-1)

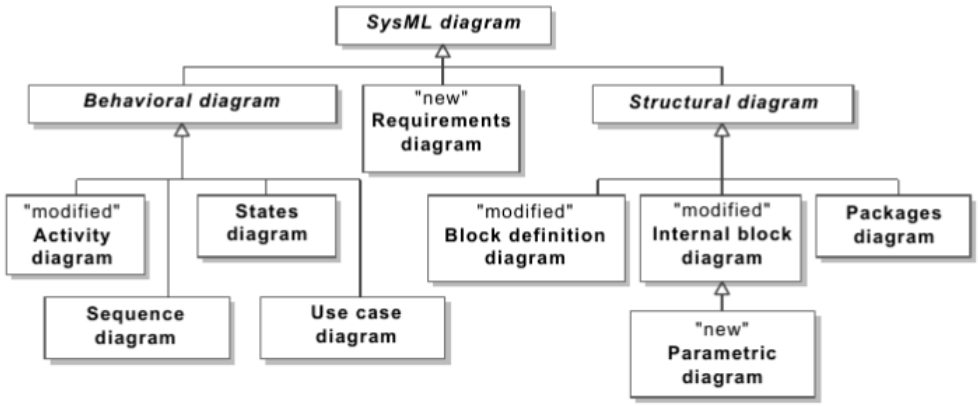


Figure 3-1 Nine Types of SysML Diagrams (Kordon, Hugues, Canals, & Dohet, 2013)

Here is a brief summary of the purpose for each type of a diagram (Delligatti, 2013):

- The **Block Definition Diagram** (BDD) is used to display elements such as blocks and value types (elements that define the types of elements that can exist in an operational system) and the relationships between those elements. Common uses for a BDD include displaying system hierarchy trees and classification trees.

- The **Internal Block Diagram** (IBD) is used to specify the internal structure of a single block. More precisely, an IBD shows the connections between the internal parts of a block and the interfaces between them.

- The **Use Case Diagram** is used to convey the use cases that a system performs and the actors that invoke and participate in them. A use case diagram is a black-box view of the services that a system performs in collaboration with its actors.

- The **Activity Diagram** is used to specify a behavior with a focus on the flow of control and the transformation of inputs into outputs through a sequence of

actions. Activity diagrams are commonly used as an analysis tool to understand and express the desired behavior of a system.

- The **Sequence Diagram** is used to specify a behavior with a focus on how the parts of a block interact with one another via operation calls and asynchronous signals. Sequence diagrams are commonly used as a detailed design tool to precisely specify a behavior as an input to the development stage of the life cycle. Sequence diagrams are also an excellent mechanism for specifying test cases.

- The **State Machine Diagram** is used to specify a behavior, with a focus on the set of states of a block and the possible transitions between those states in response to event occurrences. A state machine diagram, like a sequence diagram, is a precise specification of a block's behavior that can serve as an input to the development stage of the life cycle.

- The **Parametric Diagram** is used to express how one or more constraints specifically, equations and inequalities are bound to the properties of a system. Parametric diagrams support engineering analyses, including performance, reliability, availability, power, mass, and cost. Parametric diagrams can also be used to support trade studies of candidate physical architectures.

- The **Package Diagram** is used to display the way a model is organized in the form of a package containment hierarchy. A package diagram may also show the model elements that packages contain and the dependencies between package and model elements it contains.

- The **Requirements Diagram** is used to display text-based requirements, the relationships between requirements (containment, derive requirement, and copy), and the relationships between requirements and the other model elements that satisfy, verify, and redefine them.

Systems Engineering Modeling tools

There are several software tools solutions for systems modeling already available in the market for commercial purposes, and even some open source solutions as shown on Table 3-1.

Table 3-1 Software Tools for Systems Modeling (Kraus, Papaioannou, & Sivas, 2015)

Software Package	Creator / Publisher	License
Agilan	Visual Paradigm	Commercial
Artisan Studio	Atego	Commercial
Enterprise Architect	Sparx Systems	Commercial
Cameo Systems Modeler /	No Magic	Commercial
Rhapsody	IBM Rational	Commercial
UModel	Altova	Commercial
Modelio	Modeliosoft	Open Source
Papyrus	Atos Origin	Open Source
SysML Solution	Concept Draw	Commercial
Lattix Architect	Lattix	Commercial
Software Ideas Modeler	Dusan Rodina	Open Source
SysML Designer	ObeoNetwork	Open Source
SCADE System	Esterel Technologies	Commercial

The software tool used in this thesis project is MagicDraw from the company No Magic Inc. This software is a business process, architecture, software and system modeling tool with teamwork support. It supports UML, SysML, BPMN, and UPDM languages. This software was selected based on previous familiarity with modeling in SysML.

4. Applying Systems Thinking to VFCM Modeling

The VFCM can be considered as a "System" with multiple elements related to each other; and the result of those interactions produces emergence of functions that are used by the VFCM stakeholders. According to Crawley "there is a series of tasks that help to guide the systems thinking" (Crawley, Cameron, & Selva, 2016). For this project it was decided to follow Crawley's approach to Systems Thinking since the VFCM is a very complex system with a multiplicity of elements and complex relations between them. In this case, the development of a system model can be structured and done in an ordered manner, following the next four stages:

1. Identify the System, Form and Function
2. Identify the Entities, Form and Function
3. Identify the Relationships Among Entities
4. Analyze Emergence

In order to develop a Proof of Concept of the VFCM system model the specific part called "Features & Options" template was selected. This section has a significant complexity with hundreds of features and relations between each of them.

Identify the System, Form and Function

A Use-case diagram was created in order to analyze and understand the VFCM; this way the key stakeholders and their relations with the VFCM are exposed to discover the behavior of the VFCM system (Figure 4-1)

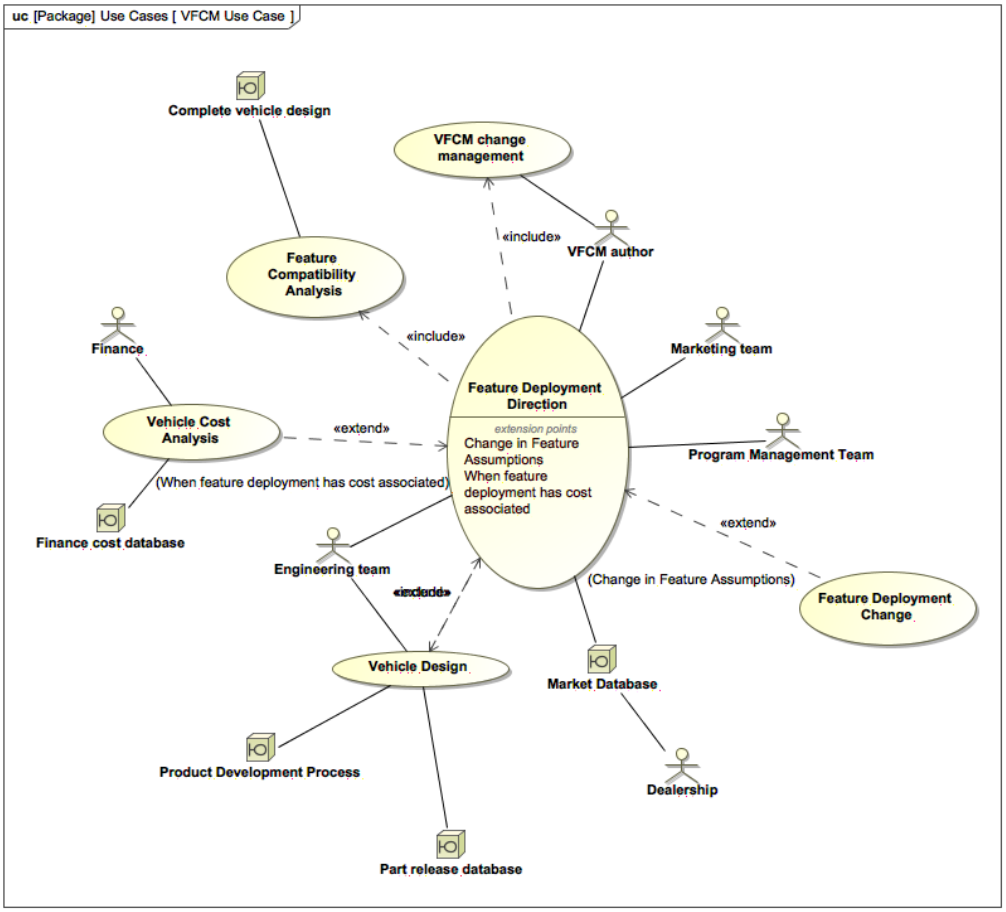


Figure 4-1 VFCM Use Case Diagram

The main Use-case shown on the diagram on Figure 4-1 is the **Feature Deployment Direction** to the program. The stakeholders interacting with this are Marketing, Engineering and Program Management teams; the VFCM is updated or modified by VFCM Author. The market needs play an important role since they are the source of vehicle user needs that need to be correlated with the deployed vehicle features contained in the VFCM. These stakeholders are suppliers of information for the VFCM, and also clients of the feature deployment information.

There are several “Include” Use-cases that are directly related with the “Feature Deployment Direction” and that are a part of the main Use-case. For example, the “Vehicle Design” is associated with the feature deployment; once a feature is approved to be deployed in the program it has to be designed within the Product Development Process and follow the component release process. Also another “Include” case is the “Feature Compatibility Analysis” that needs to be conducted to ensure every feature assumption is compatible with the rest of the vehicle features in the complete vehicle design. Likewise, the “VFCM Change Management” is an “Include” use case since the VFCM Author has to keep track of the changes and update the deployment direction accordingly.

In addition to this, the Use-case diagram shows “Extend” Use-cases that are triggered when some conditions are met. In the case of a change of the feature assumptions the “Extend” Use-case “Feature Deployment Change” is initiated, and when a feature change has any cost associated with it, the “Vehicle Cost Analysis” is started. As can be observed in the Use-case diagram, there is the Finance team as an additional stakeholder that plays role when the “Vehicle Cost Analysis” Use-case is performed, and the Finance cost database that acts as a boundary system.

When reviewing the Use-case diagram, the different behaviors of the VFCM can be observed. It can be noticed that the deployment direction requires detailed vehicle design, analysis of the compatibility of features to ensure no undesired interactions take place, and management of the VFCM to confirm consistency and

coherence of the information. Also we see that there are specific situations that activate changes in the feature deployment and cost analysis.

The VFCM Features & Options template contains a complex matrix; on one side there is a list of feature codes, each code represents a specific vehicle feature (e.g., FC557 – Music Device 1 or FC450 – Engine 1). The codes are unique and are shown organized by groups that maintain some relationship between them like all the features that compose vehicle subsystems (e.g., Suspension or Powertrain). The feature list is related to vehicle variants that can be organized by different attributes like Market, Body Style, Trim Level, Wheelbase, etc. The relations between the feature codes and the vehicle variants is the denominated Feature Deployment and explains how each independent feature connects to the variant. See Table 4-1 to visualize the concept of the existing VFCM Features and Options:

Table 4-1 VFCM Features & Options Deployment Matrix (Example)

	Vehicle Variants			
Feature List	USA Single Cab Base	USA Double Cab Luxury	…	South America Single Cab Base
Section. Suspension				
Tire 1				
Tire 2				
…				
Section. Powertrain				
Engine 1				
Engine 2				
…				

Identify the Entities, Form and Function

The base entities in the VFCM are Features and as it was mentioned before these features are identified with a **Feature Code**. For the purpose of this work a generic code format is proposed following the nomenclature "FCXXX", where "FC" means "Feature Code" and "XXX" is a consecutive number to identify generic Features to emulate a vehicle program. Each code has a series of relationships with other codes and this group of relations communicates the feature deployment direction. From the system modeling perspective each feature code can be understood as a block with certain properties inherent to itself which has links or connections to other blocks to create a network. In the VFCM the amount of relationships between feature codes is very high which makes the system really complex and hard to be shown in a simple matrix or table. See Figure 4-2 to visualize how the amount of relationships between elements can grow making the system hard to analyze and understood.

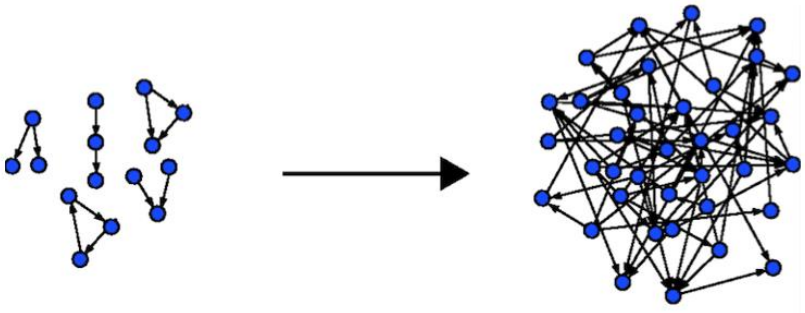


Figure 4-2 Complex Relations of Elements in a Large System

With the currently used VFCM document approach the information has to be repeated several times in the document to keep a track of the relationships no

matter what part of the document is being used. However, this can become confusing and hard to understand. See Table 4-2 with an example of a complex statement to describe relationships among feature codes. Here it is shown how the same relationship is duplicated twice in the Feature & Options matrix, it is shown at both ends of the relationship in one side as Feature FC350 contains FC184 and in the other end FC184 is included in FC350. When extrapolating this behavior throughout the VFCM the matrix becomes immense and the specification of the relationships among features displayed as text, it gets difficult to read and interpret.

Table 4-2: Example of Feature Relationships

	Feature	Feature Code	**Vehicle Variant 1**
	FR WIPER	FC350	
Relations with other features	Contains RR MIRROR 1 (FC184) when {BULKHEAD 2 (FC018), BULKHEAD 3 (FC020), BULKHEAD 5 (FC023)} is not present, and when FIXED RR WINDOW (FC462) is present, and when CAMERA 2 (FC612) is not present		Contains
	Feature	Feature Code	**Vehicle Variant 1**
	RR MIRROR	FC184	
Relations with other features	Included in FR WIPER (FC350) when {BULKHEAD 2 (FC018), BULKHEAD 3 (FC020), BULKHEAD 5 (FC023)} is not present, and when FIXED RR WINDOW (FC462) is present, and when CAMERA 2 (FC612) is not present		Includes

The SysML language offers an alternative way to represent this type of information and replace pure text descriptions with blocks and relationships among them that can be displayed graphically in multiple formats like diagrams or tables. The specification of the block and its relationships is contained in each block and all the blocks reside in a containment tree where their structural and behavioral characteristics are specified.

Blocks

The types of blocks are differentiated in the Model using "Stereotypes" which allow to define and personalize the type of blocks. Following this method, the blocks can be defined by adding information as properties associated with each stereotype. "These properties are known as tags and they are defined as properties of the stereotype block" (Holt & Perry, 2013)

There are four basic types of blocks are used in the Model (See Figure 4-3)

- **Feature Block**. These blocks are representations of each feature and they can be connected with other features to form packages of features or connected with other types of blocks to communicate the feature deployment. Associated tags:
    - Feature Code
- **Vehicle Variant Block**. This type of blocks represents a vehicle variant based on major characteristics of the vehicle such as body style, wheelbase, trim level or market. These are specified in the block specifications using tag properties. Associated tags:
    - Body Style
    - Body Style Code
    - Market
    - Market Code
    - Variant Style

- Variant Code
- Variant Name

- **Usage Block**. These blocks represent markets or groups of markets and are connected to Feature Blocks and Variants to specify deployment or restriction of features by Market. Associated tags:
  - Market
  - Market Code

- **Section Block.** These blocks are used to organize the features by subsystem so they can be grouped and uanalyzed in a set. Different types of blocks can be observed in Figure 4-3.

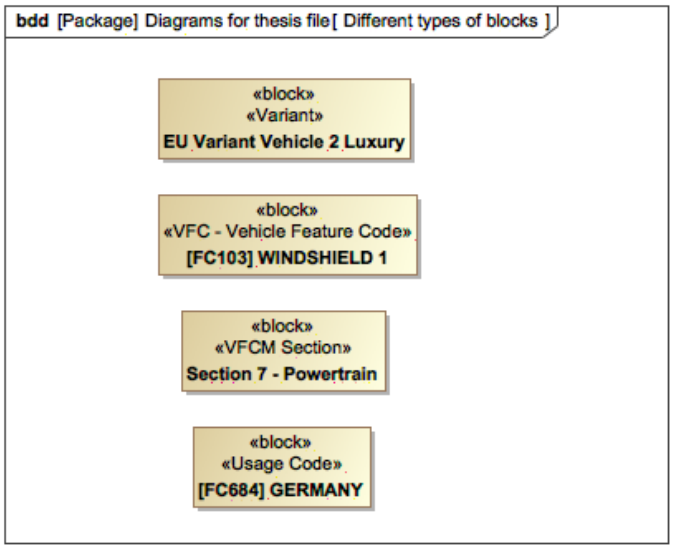


Figure 4-3 Types of Blocks in the VFCM Model (Stereotypes)

The base block used in the model is the Feature Block which represents a feature with a code and a name. This block will be connected with other feature blocks or other types of blocks such as Vehicle Variant Blocks or Usage Blocks to communicate the feature deployment. Each connection made between blocks will be added in the block specification and this will be consistent across the entire model.

Identify the Relationships Among Entities

Part Properties

The VFCM has a good number of relationships that can be represented as the block structural features called "**Part Properties**". According to Delligatti "A part property represents a structure that's internal to a block. Stated differently, a block is composed of its part properties. This relationship conveys ownership." (Delligatti, 2013). So when using Part Properties in the VFCM an Owner-Part relationship can be represented. Figure 4-4 illustrates how the Part Property can be used to replace the traditional "text-based" approach in the VFCM. Table 4-3 shows the relationships expressed as a text which was extracted from different parts of the F&O matrix; pulling together the information to track the relationships is not a simple and clear process. Instead, the model representation shows a more comprehensive view of the relation between blocks, the blocks are truly connected and this relation can be found consistently all across the model. This allows to avoid the repetition as it happens in the traditional VFCM document where the text declaration of relationships between features is repeated with different words in different document locations.

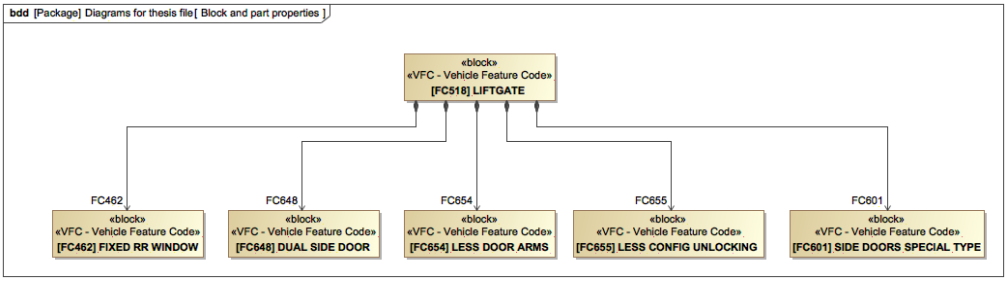


Figure 4-4 Block and Part Properties Model

Table 4-3 VFCM Contains/Includes Relationships (Extract From Different Parts of Feature and Options Matrix)

Owner	Feature Code
LIFTGATE Contains SIDE DOORS SPECIAL TYPES (FC601) Contains available DUAL SIDE DOOR (FC648) Contains LESS DOOR ARMS (FC654) Contains FIXED RR WINDOW (FC462) Contains LESS CONFIG UNLOCKING (FC655)	FC518
Parts	Feature Code
SIDE DOORS SPECIAL TYPES Included in LIFTGATE (FC518)	FC601
DUAL SIDE DOOR Included in LIFTGATE (FC518)	FC648
LESS DOOR ARMS Included in LIFTGATE (FC518)	FC654
FIXED RR WINDOW Included in LIFTGATE (FC518)	FC462
LESS CONFIG UNLOCKING Included in LIFTGATE (FC518)	FC655

"Contains" and "Includes"

In the proposed System Model, the traditional VFCM "Contains" or "Includes" relationships are represented as a Part Properties. The Part Property

relation is directional; the Owner is composed by its parts, and the parts compose the owner. In the VFCM there are hundreds of relations of this type and are commonly used to represent packs, one feature is composed or includes several others, and when you select given feature it is contained by its owner. By modeling this is SysML the relation is consistent in all the model and whenever a feature is manipulated it comes with its connected features either if they are parts or owners. See Figure 4-5 to observe how Feature in model is composed by their parts and they reside in the block specifications.

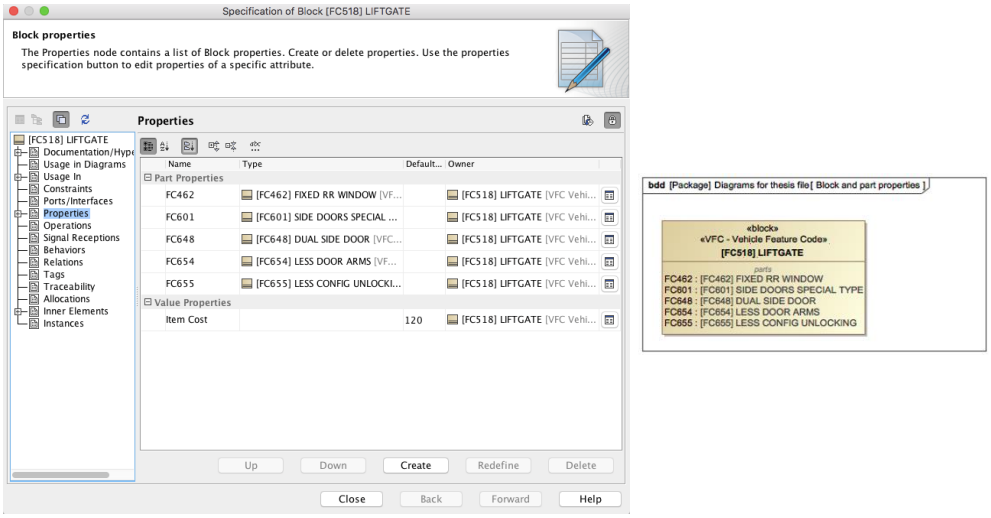


Figure 4-5 Block Specification. Part Properties

The VFCM has to communicate the availability of features for every available vehicle variant. The vehicle variant blocks in the model play the role of Owners and all its associated features are its part properties. The Features can be part properties of more than one vehicle variant. There is no limit of connections or relationships between blocks.

Optionality

There are other forms of Deployment or Optionality that are commonly used to communicate the program direction in the VFCM:

- **Standard** – Always present
- **Optional** – Optional can be or not be present depending on customer selection
- **Legal** – Obligatory by legal requirements usually associated with specific markets legislation
- **Mandatory** – Obligatory based on a strong market need or an engineering performance reason

This Optionality values are types of relationships that connect a Vehicle Variant with its features. If we refer to Table 4-1, the optionality would substitute the arrows and crosses. These values are the links that give sense to the feature deployment. When reviewing the Optionality from a SysML modeling point of view, the optionality values are variations of the type of part properties owned by the vehicle variant. So in order to represent this in the model, the forms of Optionality are defined as Stereotypes and they are applied to the Part Properties of each Vehicle Variant. This way the connection of a Vehicle Variant with a Feature Block is “Stereotyped” with Standard or Optional, etc. On Figure 4-6 a sample of a vehicle variant with some features of each type of Optionality is shown to illustrate the rationale explained before.

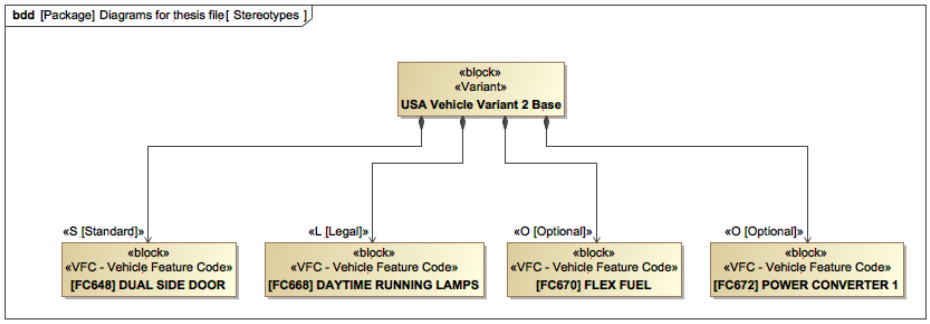
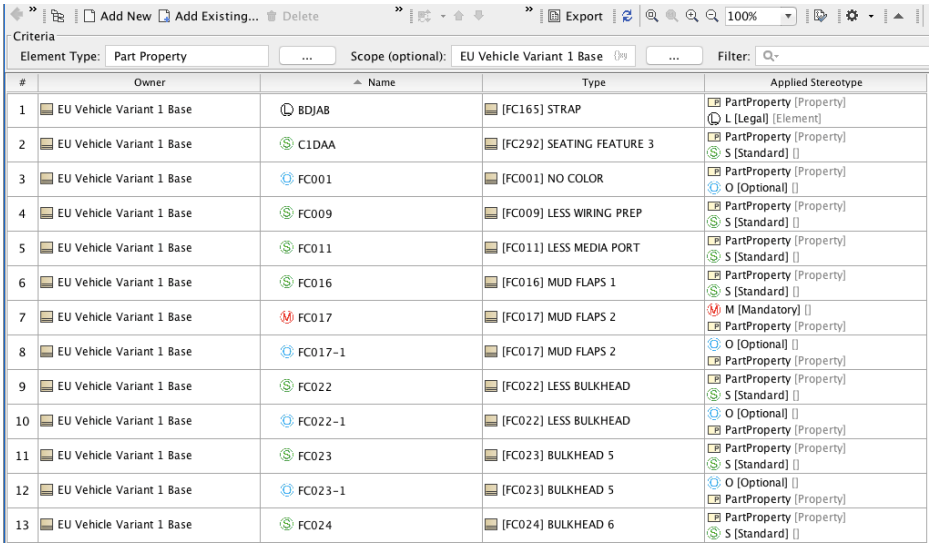


Figure 4-6 Vehicle Variant with associated Features. Example of Stereotypes

The stereotypes of Optionality have been set-up so they differentiate the part property connections and are easily identified in diagrams, tables and matrices. Various icons were added to the stereotype specifications so color and shape can help to quickly determine whether a Feature is S/O/M/L in a vehicle variant. The Figure 4-7 shows an example of a model generic table that shows the part properties owned by a vehicle variant; in this figure it can be seen how the Stereotype is shown for every Feature associated with the variant.



#	Owner	Name	Type	Applied Stereotype
1	EU Vehicle Variant 1 Base	BDJAB	[FC165] STRAP	PartProperty [Property] L [Legal] [Element]
2	EU Vehicle Variant 1 Base	C1DAA	[FC292] SEATING FEATURE 3	PartProperty [Property] S [Standard] []
3	EU Vehicle Variant 1 Base	FC001	[FC001] NO COLOR	PartProperty [Property] O [Optional] []
4	EU Vehicle Variant 1 Base	FC009	[FC009] LESS WIRING PREP	PartProperty [Property] S [Standard] []
5	EU Vehicle Variant 1 Base	FC011	[FC011] LESS MEDIA PORT	PartProperty [Property] S [Standard] []
6	EU Vehicle Variant 1 Base	FC016	[FC016] MUD FLAPS 1	PartProperty [Property] S [Standard] []
7	EU Vehicle Variant 1 Base	FC017	[FC017] MUD FLAPS 2	M [Mandatory] [] PartProperty [Property]
8	EU Vehicle Variant 1 Base	FC017-1	[FC017] MUD FLAPS 2	O [Optional] [] PartProperty [Property]
9	EU Vehicle Variant 1 Base	FC022	[FC022] LESS BULKHEAD	PartProperty [Property] S [Standard] []
10	EU Vehicle Variant 1 Base	FC022-1	[FC022] LESS BULKHEAD	O [Optional] [] PartProperty [Property]
11	EU Vehicle Variant 1 Base	FC023	[FC023] BULKHEAD 5	PartProperty [Property] S [Standard] []
12	EU Vehicle Variant 1 Base	FC023-1	[FC023] BULKHEAD 5	O [Optional] [] PartProperty [Property]
13	EU Vehicle Variant 1 Base	FC024	[FC024] BULKHEAD 6	PartProperty [Property] S [Standard] []

Figure 4-7 Vehicle Variant Part Properties Table. Optionality Stereotypes

Feature Constraints

The Features in the VFCM can have constraints that specify the way in which the features are connected. Some features can require other features in order to become feasible from and engineering point of view or consistent from a marketing point of view. Besides, some features in the vehicle variant can exclude others because of being incompatible. In the document-based VFCM these types of relationships are expressed as text strings which describe the type of a constraint and the involved features. The relation normally is directional; this means that there is a client and a supplier, and this gives meaning to the connection.

As described by Delligatti (Delligatti, 2013, p. 52), in SysML modeling the dependency is a kind of a relationship between elements, the dependency establishes a traceability between them. This type of dependency relations lets the modeling tool to perform a downstream analysis when any change is done in a feature that has dependencies, it allows to generate a list of the elements impacted by the change.

In the specific case of the VFCM model, two types of dependencies are used to make the “Requires” or “Excludes” connections between blocks. “Requires” means that other Feature needs other block in order to be available for the vehicle variant. On the other hand, “Excludes” means that another block is incompatible with the feature and, if it is present, it makes unavailable the feature for the vehicle variant. Usually the “Requires/Excludes” relationships are done from Feature Blocks to Usage Blocks (Markets) or Powertrain Features such as engines

or transmissions. Table 4-4 shows the document based approach to show require and exclude relations, while Figure 4-8 shows how it can be displayed in a Dependency Matrix of the SysML model.

Table 4-4 Requires/Excludes Relations in "Document-based" VFCM

Feature	Feature Code	Variant 1
RHD Requires {MARKET GROUP 1 (FC656), MARKET GROUP 2 (FC657), MARKET GROUP 3 (FC658)}	FC443	O*/- O[1]
HEADLAMP 5 Excludes {MARKET GROUP 2 (FC657), MARKET GROUP 4 (FC659), MARKET GROUP 3(FC658), BULGARIA (FC660), SLOVENIA (FC661), CROATIA (FC662), SLOVAKIA (FC663), MACEDONIA (FC664), ESTONIA (FC665), LATVIA (FC666), LITHUANIA (FC667)}	FC624	O*/- O[1]

Dependency Matrix is a special type of a table. It is used to create different views of custom dependencies, specifying the appearance of the matrix, that is, change the default colors of the cell and both row and column headers, assign custom icons to represent dependencies, and so forth. It also can be used to define cases when specific dependencies should be updated without rebuilding the whole matrix. One of the most important features is that the data from the Dependency Matrix can be exported in many different formats that makes it invaluable for the usage when various sources for data management and analysis are used. (Vaisnoriene, 2015)

This matrix not only facilitates management of ordinary relationships between elements, but also allows for a faster creation of traceability links between elements. This saves huge amounts of time in comparison to linking elements in

diagrams. It significantly increases applicability and usability of matrices. (Vaisnoriene, Modifying relationships in Dependency Matrix, 2015)

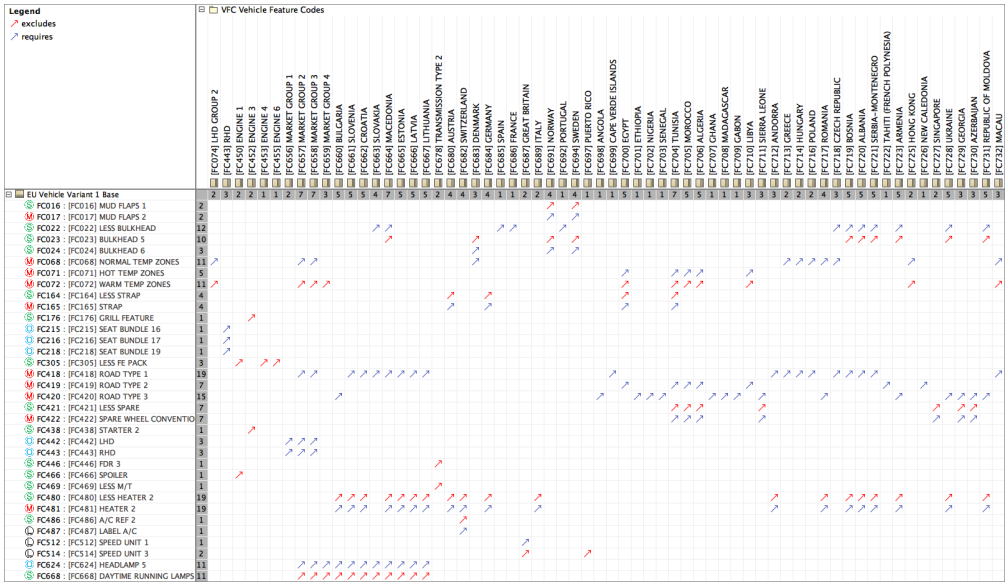


Figure 4-8 Requires / Excludes Relations in SysML Model

There is another type of Feature constraints that is commonly seen in the VFCM, it is related to the compatibility of Features, typically this happens in relations “Feature to Feature”. A feature is contained when other is present, which means that the first feature requires the second. Or it could happen in the case where one feature is contained when other is not present, which would mean that first feature excludes the second. This type of constraints is shown as complex text statements using logic connectors like “when”, “when not” or “and” as can be observed in

Table 4-5.

As opposed to this approach, the VFCM SysML model shows this type of relations using the “Requires” or “Excludes” dependency relations. Compare

Table 4-5 with Figure 4-9 to review how the same message is explained in the model without the use of complex sentences but instead a graphic explanation using a block definition diagram.

Table 4-5 Feature Constraints. “Document-based” VFCM

Feature	Feature Code	Variant 1
LESS SIDE DOORS	FC674	S*
Contains LESS 2ND ROW R/H WINDOW (FC675)		C^1
Contains LESS 2ND ROW L/H WINDOW (FC676)		C^2
Contains LESS CONFIG UNLOCKING (FC655) when LIFTGATE (FC518) is not present, and when {SPECIAL TRANSPORT PACKAGE (FC669), SPECIAL TRANSPORT PACKAGE 2 (FC671), SPECIAL TRANSPORT PACKAGE 3 (FC673)} is not present		C^3

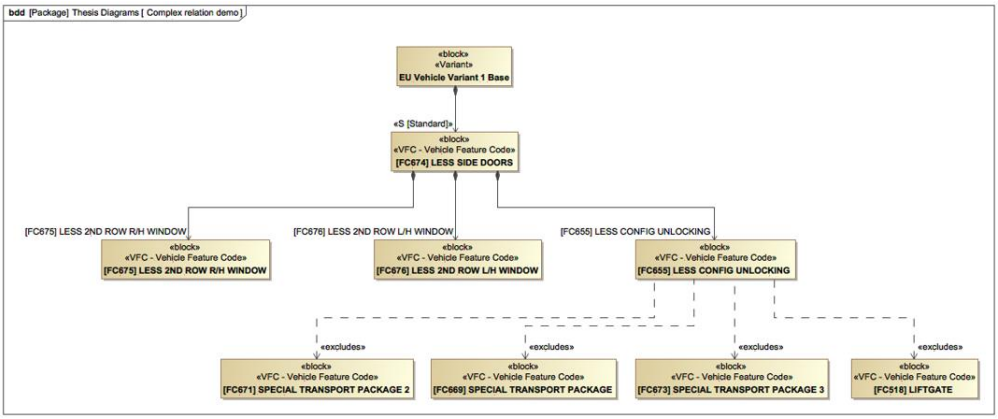


Figure 4-9 Feature Constraints. SysML based VFCM

Emergence

The principle of Emergence relates to the system behavior when the elements interact with each other to produce functionality beyond what each individual can

produce by itself. As mentioned by Crawley "the essential aspect of a system is that some new functions emerge" (Crawley, Cameron, & Selva, 2016, p. 10).

"The whole is more than the sum of the parts"

Aristotle, Metaphysics

As of now the System, Elements and Relations among them have been discussed in the context of the VFCM system. However, when the elements start interacting with each other the Emergence appears. When modeling the VFCM in SysML some of the current document-based approach concepts and principles are reproduced but all is with the objective to take advantage of the system modeling to achieve new functionality and performance and even go beyond the basic features for improved reliability, versatility and usability.

The purpose of the VFCM system model is to boost the emergence of functionality and reduce the emergency of undesired behaviors. Crawley (Crawley, Cameron, & Selva, 2016, pp. 10, 33) further points out that unanticipated or undesirable emergence is called *emergency* and that one of the goals of the systems thinking is that the emergence is understood and predictable so emergency is minimized.

Once the system elements and its relations, form and function were discussed and model techniques were developed, a bigger sample of information was modeled in order to create proof of concept of the VFCM system model behavior and emergence of functionality. Different ways to organize and present the

information were proposed in order to enhance the user experience and value obtained from the VFCM system.

The modeling effort for the VFCM proof of concept is towards looking to achieve System Elegance. Griffin (Griffin, 2010) present arguments to emphasize that there are four basic steps to achieve Elegancy on design:

- Meet Function
- Robustness
- Efficiency
- Minimization of unintended behavior.

This steps will be discussed to analyze the emergence outcomes of the proposed VFCM model.

A "Proof of Concept" VFCM System Model was developed in order to analyze and evaluate its performance when compared with the "document-based" approach. This model includes 688 feature blocks, 4 vehicle variants, 71 usage codes (Markets). This model was created in order to demonstrate how the complexity of the VFCM can be managed with the system modeling methodology. In total, the proposed model has 2,079 part property relations, 637 dependency relations and 688 value properties that creates a complex network which can serve as a scale representation of a real VFCM used in the automotive industry for a new vehicle development.

Meet Function

One of the main functions of the VFCM is to communicate the program feature assumptions with all the constraints and specifications as it has been discussed previously. Whit the use of the VFCM Model in SysML the information lives in the model containment tree and each block is unique, it contains all the specifications and relationship information with all other blocks in the model. When a single block is reviewed its impact to the whole model can be reviewed from a single specification window accessed by simple right click as can be observed in Figure 4-10. This specification window contains the list of relationships of the feature with any other blocks like other features, vehicle variants VFCM section or any constraint.

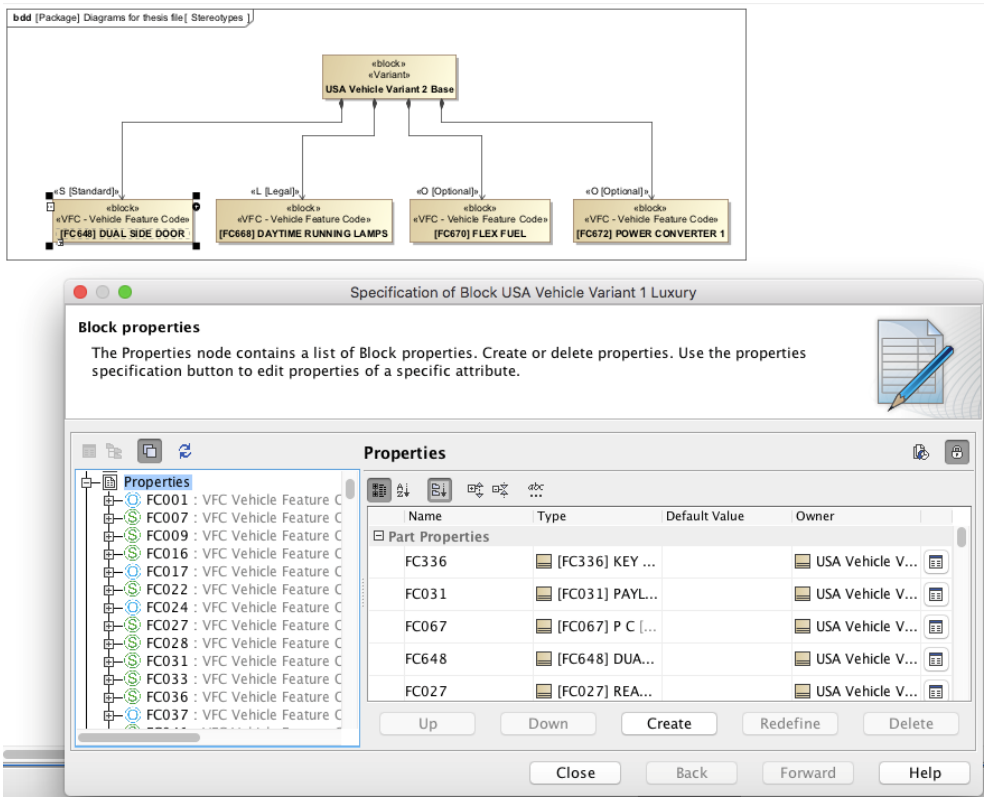


Figure 4-10 Feature Usage Analysis From Block Specification Window

Additionally, there are other ways to show the connections of a feature with other elements in the model by using the “Used by” analysis tool which shows the same information with a different level of detail and format so the user can select what works best for its particular needs (See Figure 4-11).

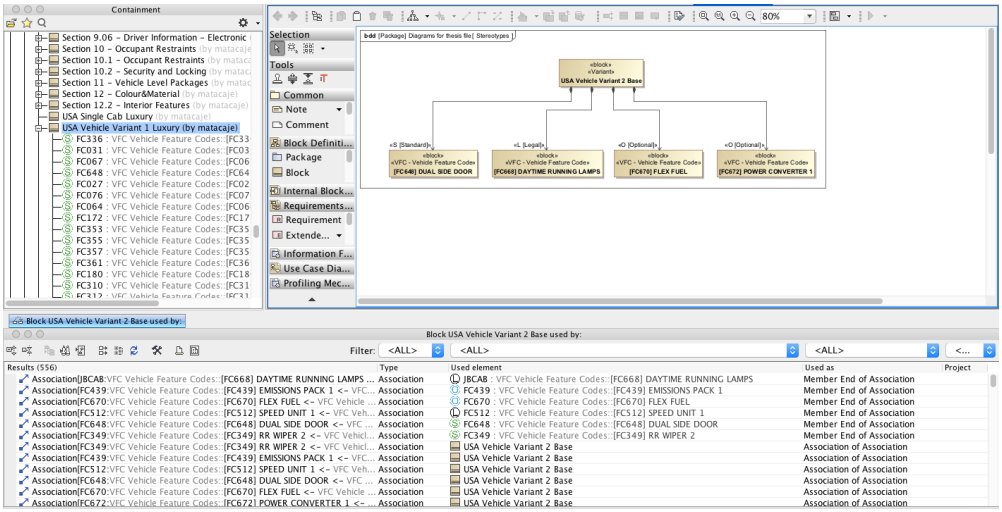


Figure 4-11 Feature Usage Analysis Using the Magic Draw "Used by" Function

Another way to analyze the elements connected to a feature block is to use the function called “Display Related Elements” that has the capability to look for all the elements connected to the feature. This tool can be customized to define the type of relation (e.g., association as part properties or dependency as “Requires” or “Excludes”), the scope of the analysis (e.g., a specific vehicle variant or the whole model) and the depth of the analysis (e.g., defining levels of relation connections to track who uses or what is related to the particular feature block analyzed). This function is useful when analyzing a feature block graphically in a Block Definition Diagram. It may be accessed with a simple right mouse button click and has a lot of power to help the user understand how the blocks are related

one to another, and also can aid in conducting analysis and making strategy decisions for feature deployment (see Figure 4-12). It can be useful when analyzing a feature change, addition or deletion; and to assess the impact of the change to the whole model or vehicle feature deployment.

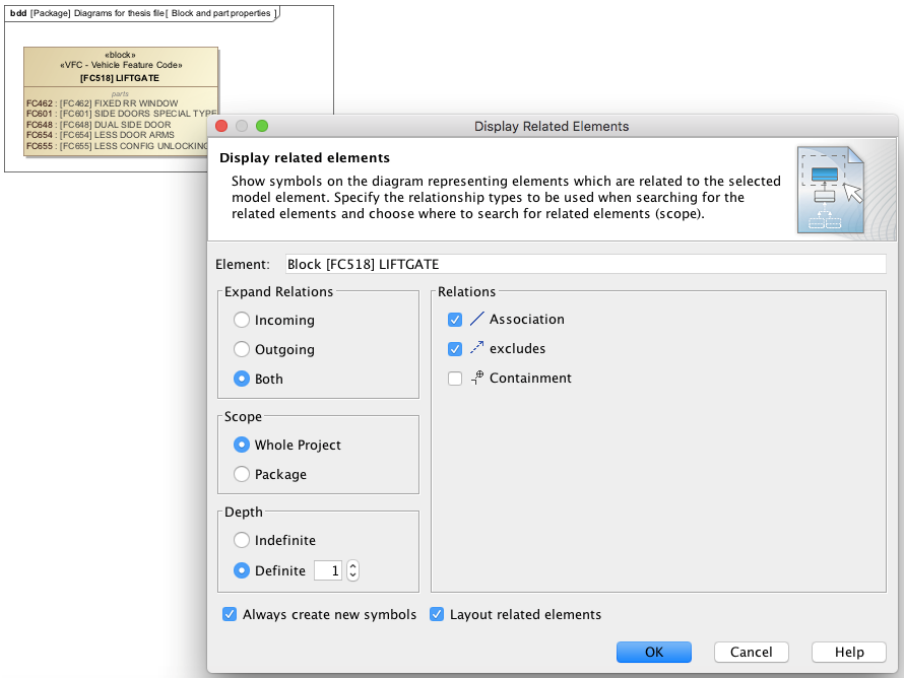


Figure 4-12: Feature Usage Analysis Using "Display Related Elements" Function

A basic feature block is shown on Figure 4-13Figure 4-13: Feature Block; this block was analyzed using the “Display Related Elements” analysis. The results of a 1-level depth analysis can be observed on Figure 4-15; the relations of this block with the immediate connected elements include its part properties and the owner of the feature analyzed. When running a 3-level depth analysis the results show a very complex network of feature blocks connected with part properties, dependency relations and it goes up to the vehicle variants and markets that use the feature analyzed (see Figure 4-15)

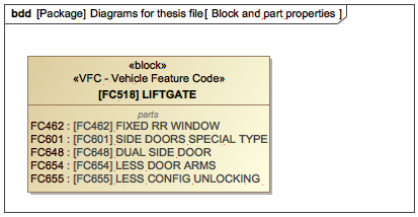


Figure 4-13: Feature Block

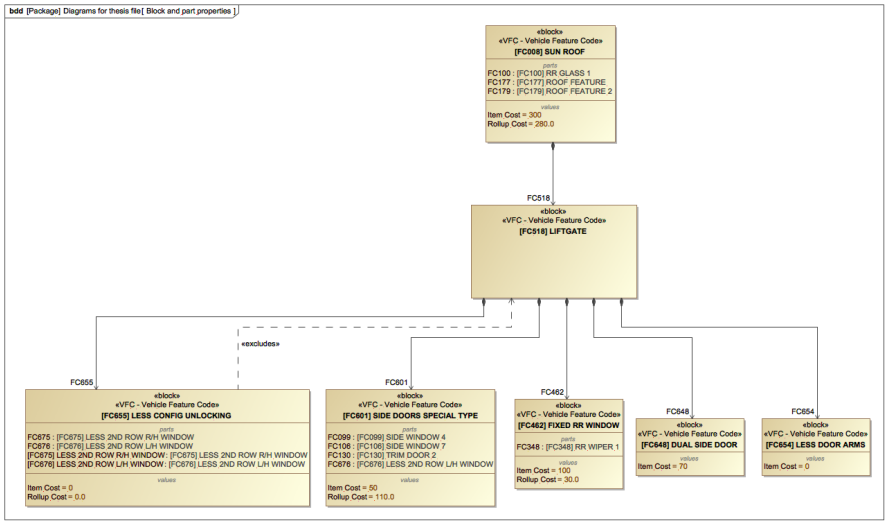


Figure 4-14: Display Related Elements Result. Depth 1-Level

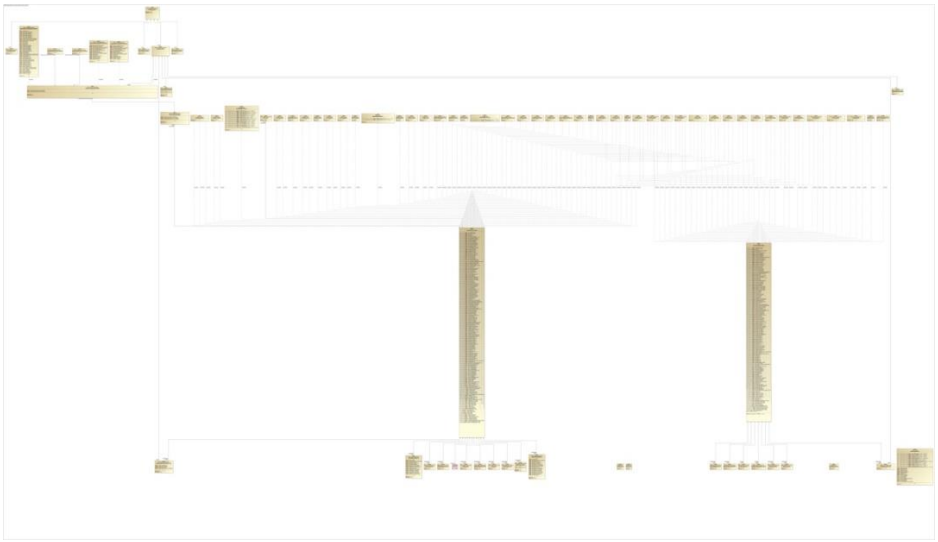


Figure 4-15: Display Related Elements Analysis result. 3 Levels

Also a variety of diagrams can be built very quickly to show the information in diverse ways so it can be interpreted. As opposed to the document-based approach where there is only one unique format template, this causes that the information of one feature is spread across the document in different sections and it is mirrored in the blocks it is connected to.

With the model based approach, the function of communicating the program feature direction can be performed more effectively since the user can personalize the way the information is displayed to analyze in further detail. Compared with reading long text descriptions including logic and conditional sentences and having to move from one place to another within the document to be able to retrieve the same message.

Robustness

Given the fact that every time a block or feature appears in the model is just a representation, and the real block is always unique in the containment tree, any change in a feature no matter in what part of the model is automatically reflected all across the model as shown on Figure 4-16, where a simple change in the feature block is propagated automatically throughout entire model updating instantly all

diagrams, tables and matrices. This allows for consistent Change Management and minimization of errors in the feature direction.

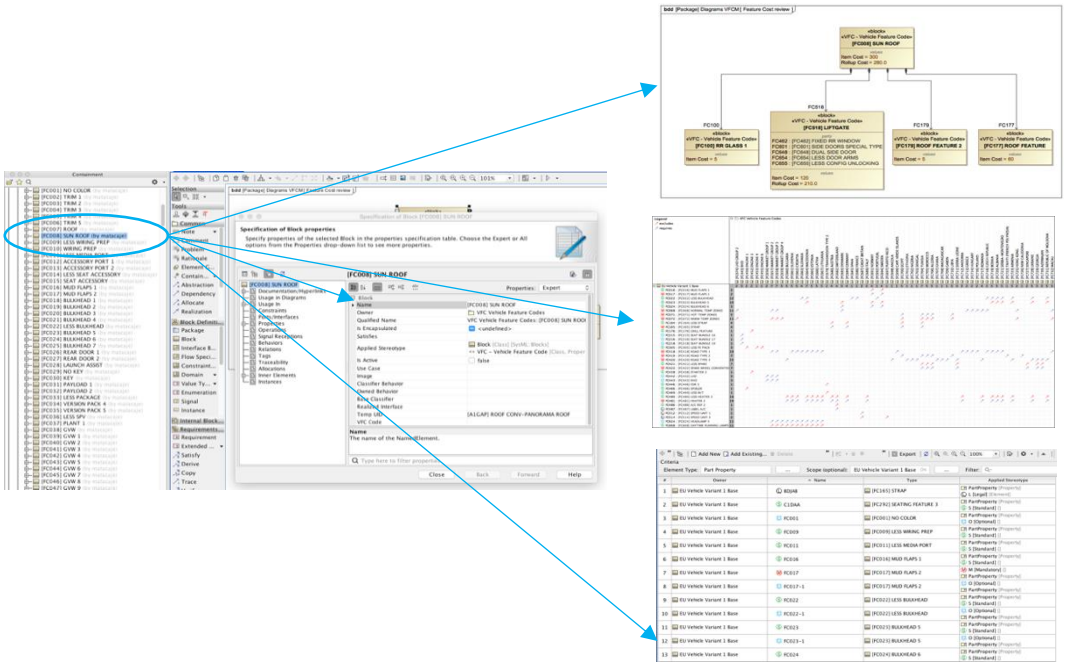


Figure 4-16: Automatic Propagation of the Feature Change in the SysML Model

In the “document-based” approach a change in one part of the document must be cascaded through the whole document manually and text has to be edited to reflect changes. Each time a feature code appears, it is a text not a model element. The changes are released in the new document versions, but it is common to discover document inconsistency much later after the document is released. This can cause waste of work

and effort and may become costly if the errors are found in the later stages of the design.

In the “model-based” approach “Metachain Operation” can be used to specify indirect (multi-level) relations between elements through the chains of the part properties. This allows to create specific tools to perform error proofing analysis such as table that automatically looks for contradictions by utilizing the “Union” and “Intersect” Boolean operations leading to a significantly reduced probability of errors; and propagate the changes immediately to the entire model if changes are made; all the related features also will reflect the change. Figure 4-17 shows an example of an Error-Proofing Table, where the tool looks for the blocks that have “Requires” and “Excludes” relationships and test if those relations contradict each other. If the result is positive it shows the features or blocks involved in the contradiction so the user can review them in details to correct the errors.

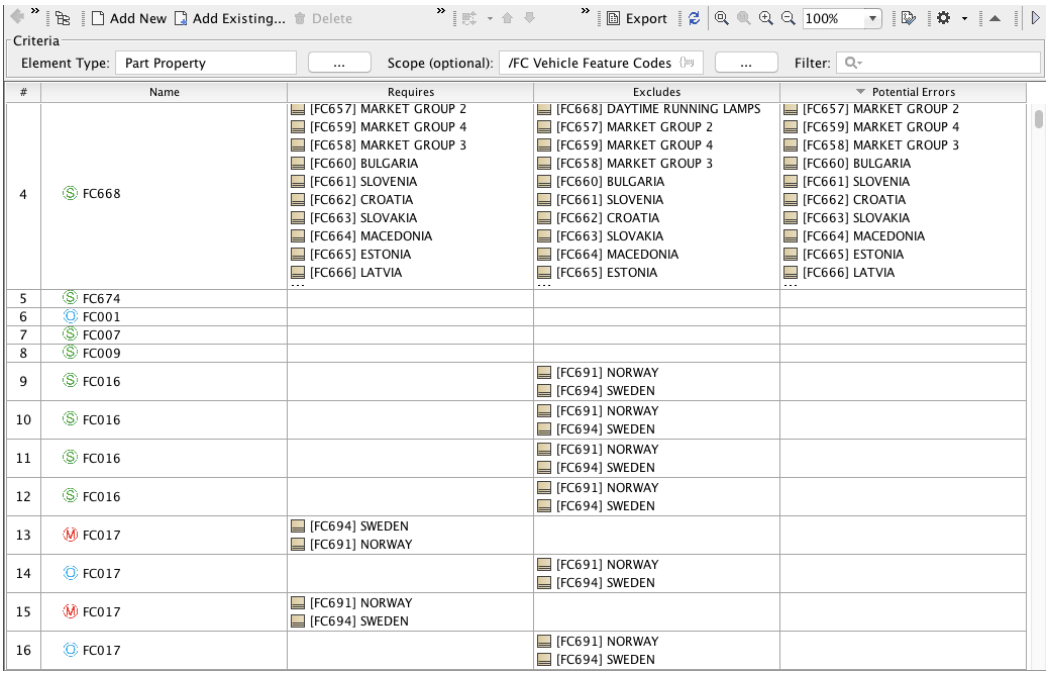


Figure 4-17 Error Proofing Tool for Requires / Excludes Relationships

This can be achieved by using the Custom Properties functionality example of which is shown on Figure 4-18.

A new Custom Property is created (in this example it is called “Potential Errors”). Then the custom Expression is built from the existing dependencies such as “Excludes” and “Requires”. This expression is used in creation of the Table shown on Figure 4-17

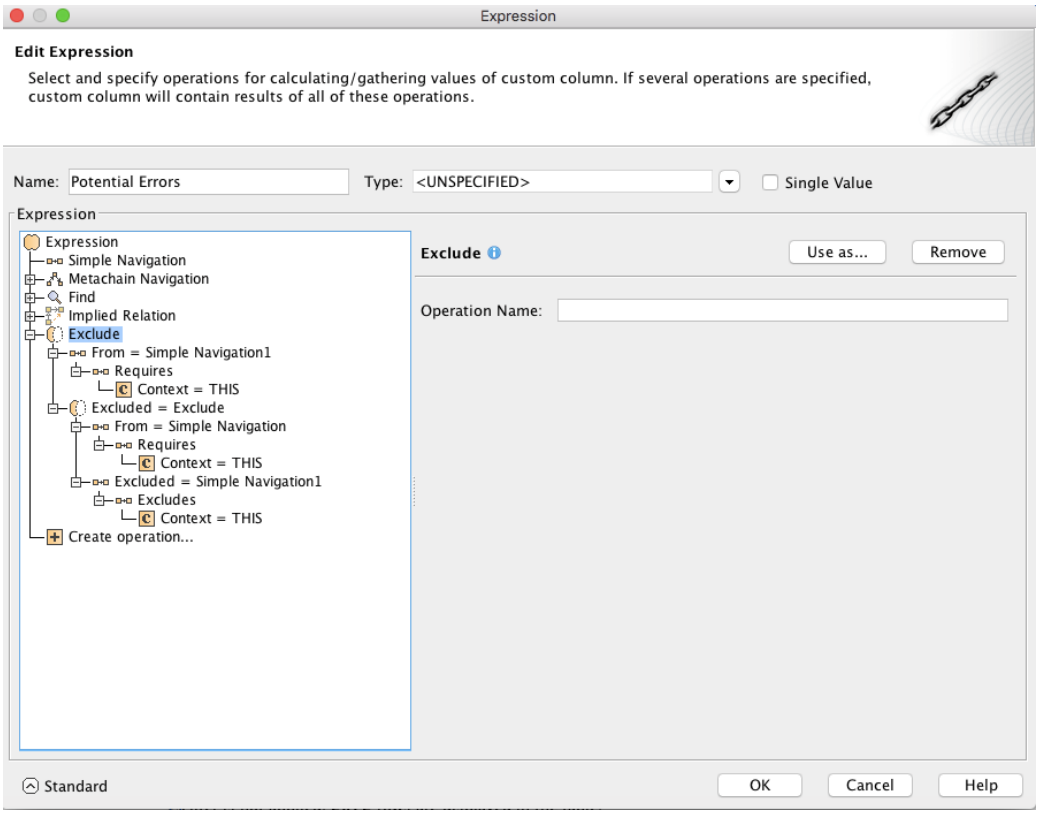


Figure 4-18 Custom Property for Usage in the Error-Proof Table Creation

Efficiency

The document-based approach is commonly more expensive than the model based approach according to Delligatti (Delligatti, 2013). The traditional VFCM requires a high volume of work to maintain, update and cascade the feature

direction; multiple meetings are required to analyze the content of the document and determine if there is any error or to ensure all team understands the document the same way so everyone can agree on the content. The described situation requires a high amount of investment in resources and time.

The SysML model is swifter to display the required information needed to make a decision. It allows the user to configure the display of the information according to the needs so it is easier to understand. Once a feature change is decided the application of the modification is immediate and automatically cascades through the model reflecting the changes in all the affected elements.

The ability to create and customize the ways in which the information is presented can also increase the efficiency of the VFCM. For example, reports that show all the variants with their Standard, Optional, Mandatory and Legal features can be created showing also the quantity of features of each type that each variant owns (See Figure 4-19). New ways to display the information can accelerate the deployment communication with the users

#	Name	Standard	Optional	Mandatory	Legal	Market	#Stand	#Opti	#Man	#Leg	# Market
1	EU Variant Vehicle 2 Luxury	FC076 : VFC Vehicle Feature Codes::[FC076] LESS MODIFICATION FC376 : VFC Vehicle Feature Codes::[FC376] AIRBAG 8 FC170 : VFC Vehicle Feature Codes::[FC170] LESS LADDER FC018 : VFC Vehicle Feature Codes::[FC018] BULKHEAD 1 FC317 : VFC Vehicle Feature Codes::[FC317] GEAR KNOB 1 FC591 : VFC Vehicle Feature Codes::[FC591] PHONE INTERFACE 1 FC486 : VFC Vehicle Feature Codes::[FC486] A/C REF 2 FC459 : VFC Vehicle Feature Codes::[FC459] BRAKE LIGHT FC019 : VFC Vehicle Feature Codes::[FC019] BULKHEAD 2 FC409 : VFC Vehicle Feature Codes::[FC409] TIRE BRAND ...	FC066 FC037 FC412 FC605 FC616 FC335 FC104 FC021 FC427 FC481 ...	FC418 FC419 G2AAF FC068 FC071 FC165 D17AD FC072 FC017 FC420 ...	FC487 BDJAB : JBCAB : FC514 FC512	[FC657] MARKET GROUP 2 [FC659] MARKET GROUP 4 [FC658] MARKET GROUP 3 [FC660] BULGARIA [FC661] SLOVENIA [FC662] CROATIA [FC663] SLOVAKIA [FC664] MACEDONIA [FC665] ESTONIA [FC666] LATVIA ...	140	50	11	5	60
2	EU Vehicle Variant 1 Base	FC635 : VFC Vehicle Feature Codes::[FC635] LIGHT FEATURE 4 FC194 : VFC Vehicle Feature Codes::[FC194] MIRROR COLOR 3 FC089 : VFC Vehicle Feature Codes::[FC089] LESS WINDOW FEATURE FC449 : VFC Vehicle Feature Codes::[FC449] EMISSIONS PACK 4 FC617 : VFC Vehicle Feature Codes::[FC617] LESS POD FC355 : VFC Vehicle Feature Codes::[FC355] RR BUMPER 2 FC591 : VFC Vehicle Feature Codes::[FC591] PHONE INTERFACE 1 FC337 : VFC Vehicle Feature Codes::[FC337] KEY FREQ 2 FC457 : VFC Vehicle Feature Codes::[FC457] BRAKE FEATURE 1 FC284 : VFC Vehicle Feature Codes::[FC284] LESS RR ROW 3 ...	FC422- FC195 FC077 FC616 FC442 FC411 FC435 FC650 FC443 FC531 ...	FC072 FC420 FC422 FC481 FC017 FC419 FC165 FC068 FC071 FC418 ...	FC668 FC487 FC514 FC512 BDJAB : ...	[FC694] SWEDEN [FC691] NORWAY [FC074] LHD GROUP 2 [FC657] MARKET GROUP 2 [FC658] MARKET GROUP 3 [FC683] DENMARK [FC713] GREECE [FC714] HUNGARY [FC716] POLAND [FC717] ROMANIA ...	155	39	10	5	58
3	USA Vehicle Variant 1 Luxury	FC064 : VFC Vehicle Feature Codes::[FC064] PAINT 1 FC457 : VFC Vehicle Feature Codes::[FC457] BRAKE FEATURE 1 FC093 : VFC Vehicle Feature Codes::[FC093] WINDOW FEATURE 4 FC306 : VFC Vehicle Feature Codes::[FC306] FE PACK 1 FC642 : VFC Vehicle Feature Codes::[FC642] STOP LAMP FC126 : VFC Vehicle Feature Codes::[FC126] LESS SEAT FEATURE FC270 : VFC Vehicle Feature Codes::[FC270] LESS STOWAGE FC444 : VFC Vehicle Feature Codes::[FC444] FDR 1 FC466 : VFC Vehicle Feature Codes::[FC466] SPOILER FC160 : VFC Vehicle Feature Codes::[FC160] LESS LOAD FLOOR ...	FC177 FC668 FC057 FC439 FC679 FC431 FC548 FC179 FC024 FC672 ...	FC072 FC068 FC418 ...	FC512 FC458 FC487 ...	[FC455] ENGINE 6 [FC678] TRANSMISSION TYPE 2 [FC452] ENGINE 3 [FC679] TRANSMISSION TYPE 3 [FC453] ENGINE 4 [FC695] USA [FC697] PUERTO RICO ...	148	45	3	3	7
4	USA Vehicle Variant 2 Base	FC476 : VFC Vehicle Feature Codes::[FC476] DUCT FC530 : VFC Vehicle Feature Codes::[FC530] FEATURE X 3 FC603 : VFC Vehicle Feature Codes::[FC603] POWER OUTLET 2 FC311 : VFC Vehicle Feature Codes::[FC311] DOOR HANDLE 3 FC376 : VFC Vehicle Feature Codes::[FC376] AIRBAG 8 FC626 : VFC Vehicle Feature Codes::[FC626] HEADLAMP 7 FC051 : VFC Vehicle Feature Codes::[FC051] LESS UPGRADE FEATURE FC164 : VFC Vehicle Feature Codes::[FC164] LESS STRAP FC463 : VFC Vehicle Feature Codes::[FC463] TRACKING 1 FC181 : VFC Vehicle Feature Codes::[FC181] WIDE BODYSIDE MLDG- ...	FC464 FC082 FC650 FC653 FC455 FC001 FC080 FC670 FC592 FC439 ...	FC072 FC068 FC070 ...	FC513 FC512 JBCAB : ...	[FC455] ENGINE 6 [FC678] TRANSMISSION TYPE 2 [FC452] ENGINE 3 [FC679] TRANSMISSION TYPE 3 [FC453] ENGINE 4 [FC697] PUERTO RICO [FC695] USA [FC696] CANADA	130	41	3	3	8

Figure 4-19: Vehicle Variants Feature and Markets Summary Table

In addition to this, MagicDraw tool is very flexible and efficient to exchange information with MS Excel using *.csv or *.xls(x) file formats. MagicDraw uses **Import** command such as Import from → CSV File or Import → Excel/CSV File which allows to extract information from a traditional Excel table that contains information about blocks, properties, dependencies or any other type of model relation. It can be configured to adapt to the modeler needs so large amount of data can be imported quickly to create multiple blocks or relations in a few seconds. This also offers a link of compatibility between existing "document-based" VFCM and "model-based" VFCM since document information can be formatted to be imported to the model without much effort.

Table 4-6 shows a sample import of .CSV template with information required in the VFCM model including "Owner Block" and all its "Part Properties". This template can be used to import thousands of part property relations at once. The

Table 4-6 can be compared with the corresponding model output after the import as shown on Figure 4-20.

Figure 4-21 and Figure 4-22 show the .CSV Import Tool and column selection interface in MagicDraw with variety of options for import of multiple model element specifications. The vast range of options to select the data to be imported represents a powerful tool which can improve the efficiency greatly when adding high amount of information to the model.

Table 4-6: Sample .CSV table to import Feature Block Part Properties

Owner Block	Part Property Block
[FC200] SEAT BUNDLE 1	[FC002] TRIM 1
[FC200] SEAT BUNDLE 1	[FC014] LESS SEAT ACCESSORY
[FC200] SEAT BUNDLE 1	[FC197] SEAT BAG 2
[FC200] SEAT BUNDLE 1	[FC198] LESS SEAT BAG
[FC200] SEAT BUNDLE 1	[FC251] LESS D ARMREST
[FC200] SEAT BUNDLE 1	[FC253] LESS P ARMREST
[FC200] SEAT BUNDLE 1	[FC258] SEAT POSITIONER 5
[FC200] SEAT BUNDLE 1	[FC262] LESS TRAY
[FC200] SEAT BUNDLE 1	[FC267] SEAT ADJUSTER 2
[FC200] SEAT BUNDLE 1	[FC268] SEAT ADJUSTER 3
[FC200] SEAT BUNDLE 1	[FC272] SEAT FEATURE 3
[FC200] SEAT BUNDLE 1	[FC274] SEAT FEATURE 3
[FC200] SEAT BUNDLE 1	[FC288] AUTO ADJUSTER 2

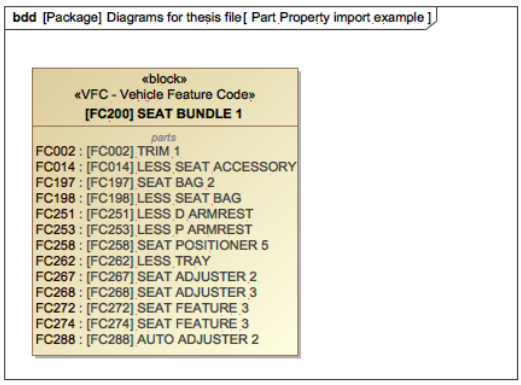


Figure 4-20: Part Properties imported using Magic Draw .CSV import tool

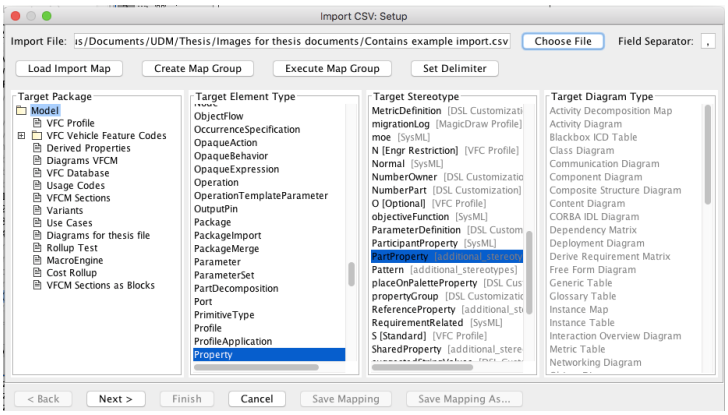


Figure 4-21: Magic Draw "Import from .CSV" Tool Interface

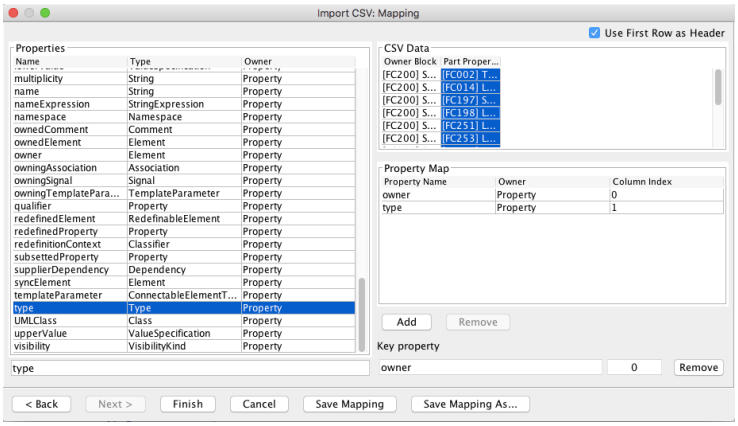


Figure 4-22: Magic Draw "Import from .CSV" Column Selection Screen

Moreover, the .CSV import function offers a capability to support establishment of work teams with experienced and non-experienced modelers that can work together to develop pilot modeling projects, merge their own expertise, and build up modeling skills together.

Minimization of Undesired Behavior

With the use of the SysML Model the undesired behavior is minimized greatly, the characteristics and the capabilities of the system model allow to

mitigate common errors on document vehicle feature matrices. The following undesired behaviors are addressed:

Inconsistency and contradictions in feature deployment

Inconsistency of data or contradictions in the feature deployment is reduced because the features are connected with each other the information is all linked and the errors can pop-up easier than in a document feature matrix. As described in the Robustness section the model gives flexibility to generate error-proofing tables or diagrams to automatically look for suspicious data throughout the model. Besides this, the ability to automatically update any change across the model reduces the potential of maintaining old information that is no longer valid.

Complexity in the Document

The "document-based" approach can become very complex when explaining the relations among the features by using just text, this is a frequent complaint of stakeholders when using the current VFCM. The model approach has the capability of deploying the information in diverse ways so the message can be easily understood by the users either by looking at a matrix, a diagram or a table explaining the relation among the features. The described tools can be personalized and adjusted to reduce or increase the scope so the amount of information is manageable and usable.

5. New VFCM Functionality. Cost Rollup

After the model had been setup as described in the previous chapter as a part of the Proof of Concept, the following functionality was developed in order to

take advantage of the model capabilities and the availability of information and block connections.

One of the main drivers of the decision making for feature deployment is Cost; adding or removing features from the program assumptions have an enormous impact in the overall vehicle production cost and also on the projected revenues. With the traditional document approach the cost analysis is done independently as a confirmation after the feature deployment is released in the VFCM. This consumes time and resources and it is common to find issues and come back to make updates to the original vehicle feature deployment after the cost analysis results are ready.

Based on this, it is proposed to include a cost rollup analysis in the VFCM model to include feature cost as a value property of each block. The value properties are defined in each element specifications and each element is connected to others, they can belong to packages, require or exclude other features and all together integrate into the Vehicle Variants. A sample cost database was generated to demonstrate the use of the cost rollup analysis (see Figure 5-1).

#	Owner	Default Value	Feature Cost [Dollars]
70	[FC041] GVW 3	0	0
71	[FC440] EMISSIONS PACK 2	600	600
72	[FC632] LIGHT FEATURE 1	5	5
73	[FC219] SEAT BUNDLE 20	220	220
74	[FC084] REPAIR KIT	5	5
75	[FC166] LESS GRAB HANDLE	0	0
76	[FC451] ENGINE 2	3000	3000
77	[FC125] CUP HOLDER	8	8
78	[FC149] LESS HOOK	0	0
79	[FC535] FEATURE X 8	100	100
80	[FC343] DOOR LOCKS 5	10	10
81	[FC248] SEAT BUNDLE 49	300	300
82	[FC146] DOOR HANDLE 2	20	20
83	[FC003] TRIM 2	5	5
84	[FC417] LUG NUT 2	5	5
85	[FC284] LESS RR ROW 3	0	0
86	[FC523] SCREEN 1	0	0
87	[FC053] UPGRADE FEATURE P...	250	250
88	[FC117] GLOVE BOX 1	0	0
89	[FC283] RR ROW 2	10.0	10.0
90	[FC202] SEAT BUNDLE 3	320	320
91	[FC445] FDR 2	5	5
92	[FC311] DOOR HANDLE 3	5	5
93	[FC273] LESS SEAT FEATURE 3	0	0
94	[FC470] M/T	2000	2000
95	[FC193] MIRROR COLOR 2	5	5
96	[FC446] FDR 3	5	5
97	[FC352] JETS	20	20

Figure 5-1: Features Block Value Properties (Sample Cost Database)

The Cost Roll-up Analysis was performed by using a macro that reads the value properties of the target block and looks for the value properties of all the blocks connected to it. This can be reproduced at the feature package level or at the whole variant level. It basically conducts a sum of the value properties of each block connected to the target block. In this particular case the value property created was a Feature Cost, so the macro creates a cost roll-up value for each block that contains parts and this is repeated all the way up to the top feature as shown on Figure 5-2.

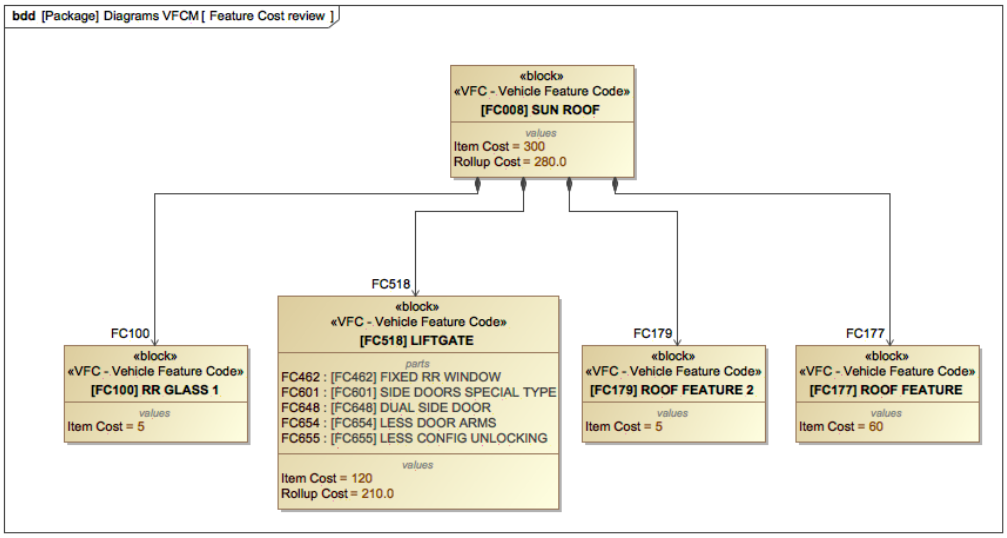


Figure 5-2: BDD for Feature Cost and Cost Roll-up Analysis

This macro allows to conduct a roll-up calculation to determine the cost of adding or removing a feature, the cost of packages and the overall cost of the whole vehicle. It can also be very useful to support management decisions whether approve or reject a change right away when reviewing the VFCM model. The capability of this analysis can expand to calculate the cost of the vehicle variants (see Figure 5-3) so the users can analyze immediately how the different trim levels compare with each other to setup strategies for marketing and costing. The same type of macro can be applied for other analysis such as Vehicle Weight Roll-up that has the potential to expand the capabilities of the VFCM model even further.

#	Owner	Default Value	Type
1	USA Vehicle Variant 2 Base	16570.0	cost [dollars]
2	USA Vehicle Variant 1 Luxury	19766.0	cost [dollars]
3	EU Variant Vehicle 2 Luxury	19598.0	cost [dollars]
4	EU Vehicle Variant 1 Base	10186.0	cost [dollars]

Figure 5-3: Cost Rollup Analysis Result for Complete Vehicle Variants

6. *Next Steps and Further Research Opportunities*

As Systems Engineering and the related to the discipline tools become more widely accepted in the Automotive World, SysML model approach becomes a game-changer on how the automotive complexity management and system modeling are handled. SysML modeling is a unique and universal approach and it can be used not only for the complexity management as described in this research but also for System Architecture such as electrical systems as those become the most complex systems in the automotive industry. SysML provides vast range of the reports and validation tools which are crucial for reducing errors and increasing the efficiency of the data management.

We believe that the results obtained in this research are a solid ground for further investigation of the opportunities for parametric SysML approach usage in creating, maintaining and managing the Vehicle Feature Code Matrix.

Next Steps would include but not be limited to:

- Widen the scope of the model to add specific model analysis tools to engineering teams like Powertrain or Electrical

- Research among diverse OEMs formats and compile a database of necessary inputs to the model
- Define more accurate requirements and perform Requirement Model Analysis as suggested on the Figure 6-1

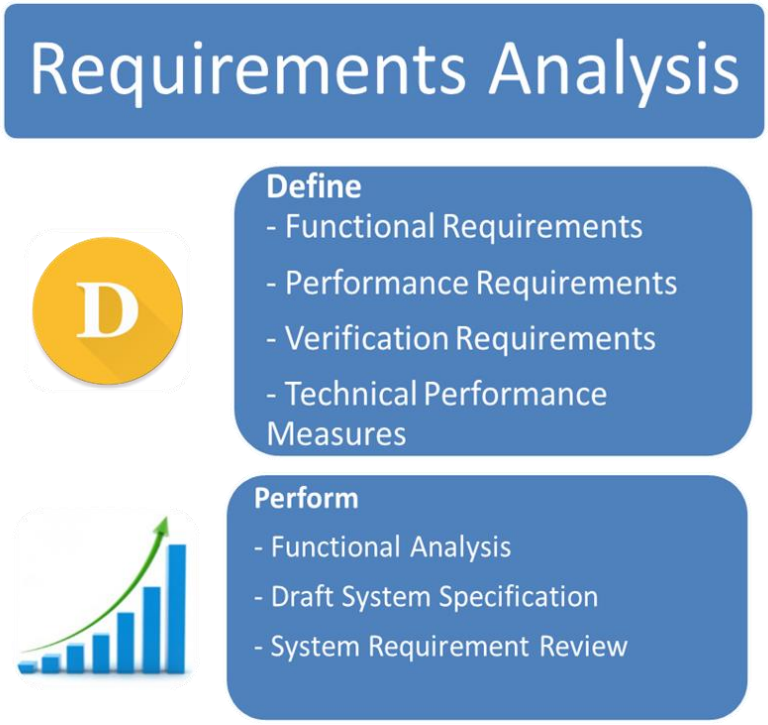


Figure 6-1 Next Steps. Requirement Analysis

- Research and develop new output tables and reports to satisfy the needs of the stakeholders

7. Conclusions and Recommendations for Further Development

We have reviewed the advantages of the system modeling compared with the "document-based" approach for the Vehicle Feature Complexity Matrix. The Proof of Concept presented in this research has demonstrated the potential to meet the function of representing and communicating the vehicle program feature deployment direction to the interested parties, increase the efficiency, robustness and minimize the undesired behaviors in the VFCM.

The VFCM model can improve the consistency of the information, reduce the amount of resources dedicated to maintain and update the feature directions, and provide a greater capability to display the information in tables, matrices and diverse diagrams that can also be customized to define the scope of the analysis, so the user looks specifically at what is needed in order to make an informed decision, evaluate or propose a feature change.

The system modeling of the VFCM opens new possibilities to expand the current functionality to integrate other processes as we have reviewed with the Cost Roll-up Analysis which uses the existing model to produce new valuable information for the stakeholders. Beyond what we have discussed in this project there is still a wider field to expand the capabilities for creating new reports or model tools to attend to specific needs of VFCM users like functional or engineering attribute teams, marketing and finance teams.

The modeling tool used in this thesis project was Magic Draw. It proved to be very capable and robust to manage big and complex networks of highly interconnected blocks as reviewed in this project. The software interface is convenient and highly customizable to be able to import information easily, build diagrams, tables and matrices very quickly and efficiently. This modeling tool offers great advantages to manage the VFCM proof of concept model and we consider that its capability can be extrapolated to a larger scale to VFCM models in real life automotive vehicle programs.

As a further step to continue this development a joint project with an OEM would be required in order to launch a pilot model using real vehicle program information, getting interaction with the users and feedback from all the stakeholders. This will help to apply this Proof of Concept to a real world application and to make a deeper evaluation of the effectiveness and the benefits of VFCM system model.

a. References

Collins-Dictionary. (2016, November 5). *http://www.collinsdictionary.com*. Retrieved from Collins Dictionary: http://www.collinsdictionary.com/dictionary/english/model

Crawley, E., Cameron, B., & Selva, D. (2016). *System Architecture. Strategy and Product Development for Complex Systems.* Hoboken, NJ, USA: Pearson.

Delligatti, L. (2013). *SysML Distilled. A Brief Guide to the Systems Modeling Language.* Crawfordsville, Indiana, USA: Addison-Wesley.

Deployment, Q. (2005). *Kano Model Analysis.* Retrieved from UCALGARY.CA: http://people.ucalgary.ca/~design/engg251/First%20Year%20Files/kano.pdf

Faulconbridge, I. a. (2002). Managing Complex Technical Projects. Norwood, USA.

Griffin, M. (2010). How do we fix System Engineering? *61st International Astronautical Congress.* Prague, Czech Republic.

H. Schleich, J. S. (2007). *MANAGING COMPLEXITY IN AUTOMOTIVE PRODUCTION.* Retrieved from Labnexo.com: http://www.labnexo.com/wp-content/uploads/2013/08/pdf-n%C2%BA80-Managing-complexity-in-automotive-production.-2007.pdf

Holt, J., & Perry, S. (2013). *SysML for Systems Engineering. 2nd Edition : A Model-Based Approach* (2nd ed.). Stevenage, UK: Institution of Engineering and Technology.

Kordon, F., Hugues, J., Canals, A., & Dohet, A. (2013). *Embedded Systems. analysis and Modeling with SysML, UML and AADL* (1st ed.). Hoboken, NJ, USA: Wiley.

Kossiakof, A., Sweet, W., Seymour, S., & Biemer, S. (2011). *Systems Engineering Principles and Practice* (2nd ed.). Hoboken, New Jersey, USA: Wiley.

Kraus, R., Papaioannou, G., & Sivas, A. (2015). *Application of Model Based System Engineering (MDSE) Principles to an Automotive Driveline Sub-System Architecture (Master's Dissertation).* Detroit, MI, USA: University of Detroit Mercy.

R. Ian Faulconbridge, M. J. (2003). Managing Complex Technical Projects: A Systems Engineering Approach. In M. J. R. Ian Faulconbridge, *Managing Complex Technical Projects: A Systems Engineering Approach* (p. 29). Norwood, MA: ARTECH House, Inc.

Vaisnoriene, D. (2015, July 5). *Dependency Matrix*. Retrieved from No Magic Documentation: http://docs.nomagic.com/display/MD182/Dependency+matrix

Vaisnoriene, D. (2015, August 26). *Modifying relationships in Dependency Matrix*. Retrieved from MagicDraw 18.4 Documentation: http://docs.nomagic.com/display/MD184/Modifying+relationships+in+Dependency+Matrix

Weiss, S. I. (2013). Product and Systems Development: A Value Approach. Somerset , USA.

APPENDIX A. Figures and Tables

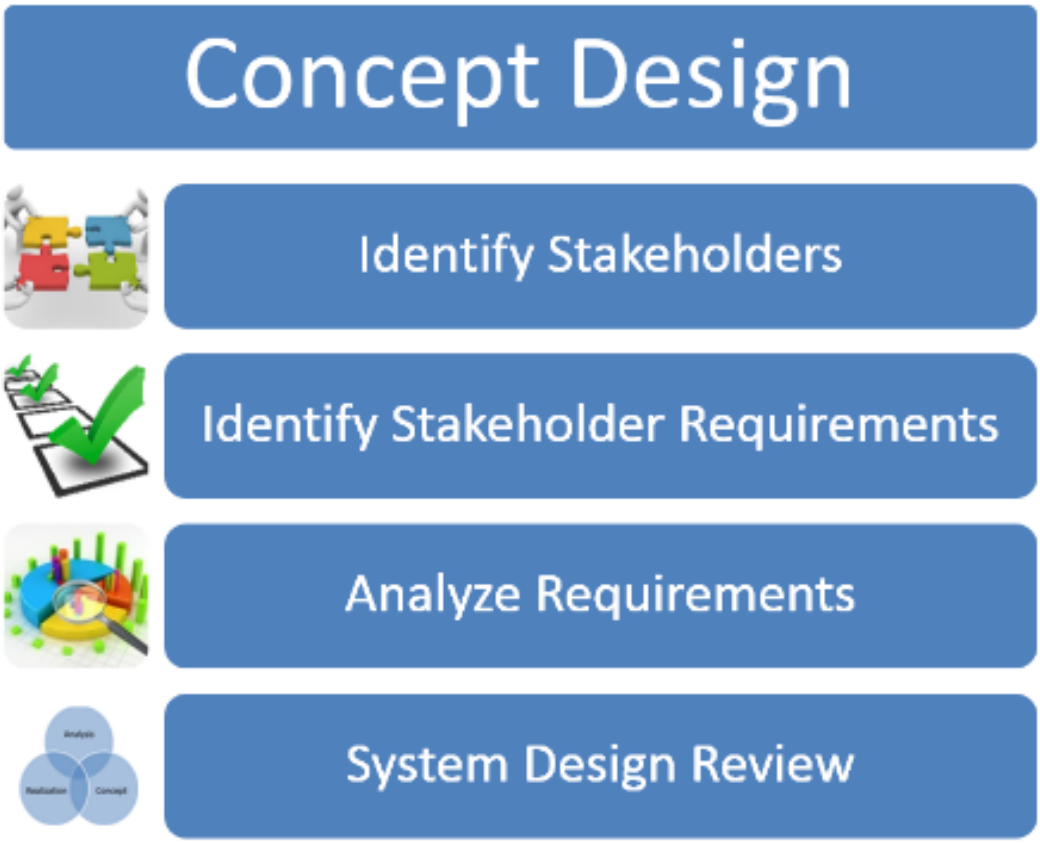


Figure 2-1 Concept Design Stages

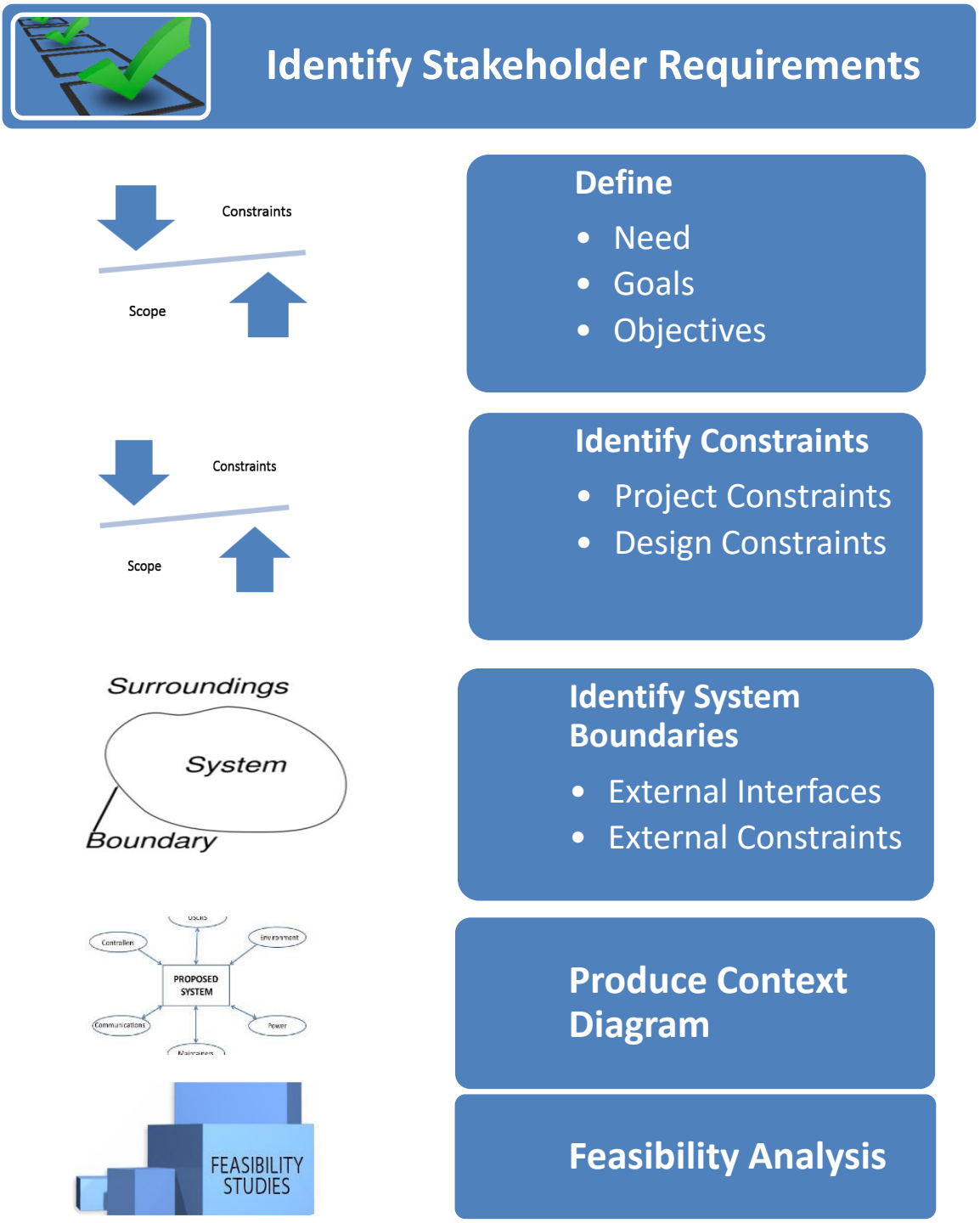


Figure 2-2 Steps to Identify Stakeholder Requirements

Figure 2-3 Stakeholders Needs, Goals and Objectives

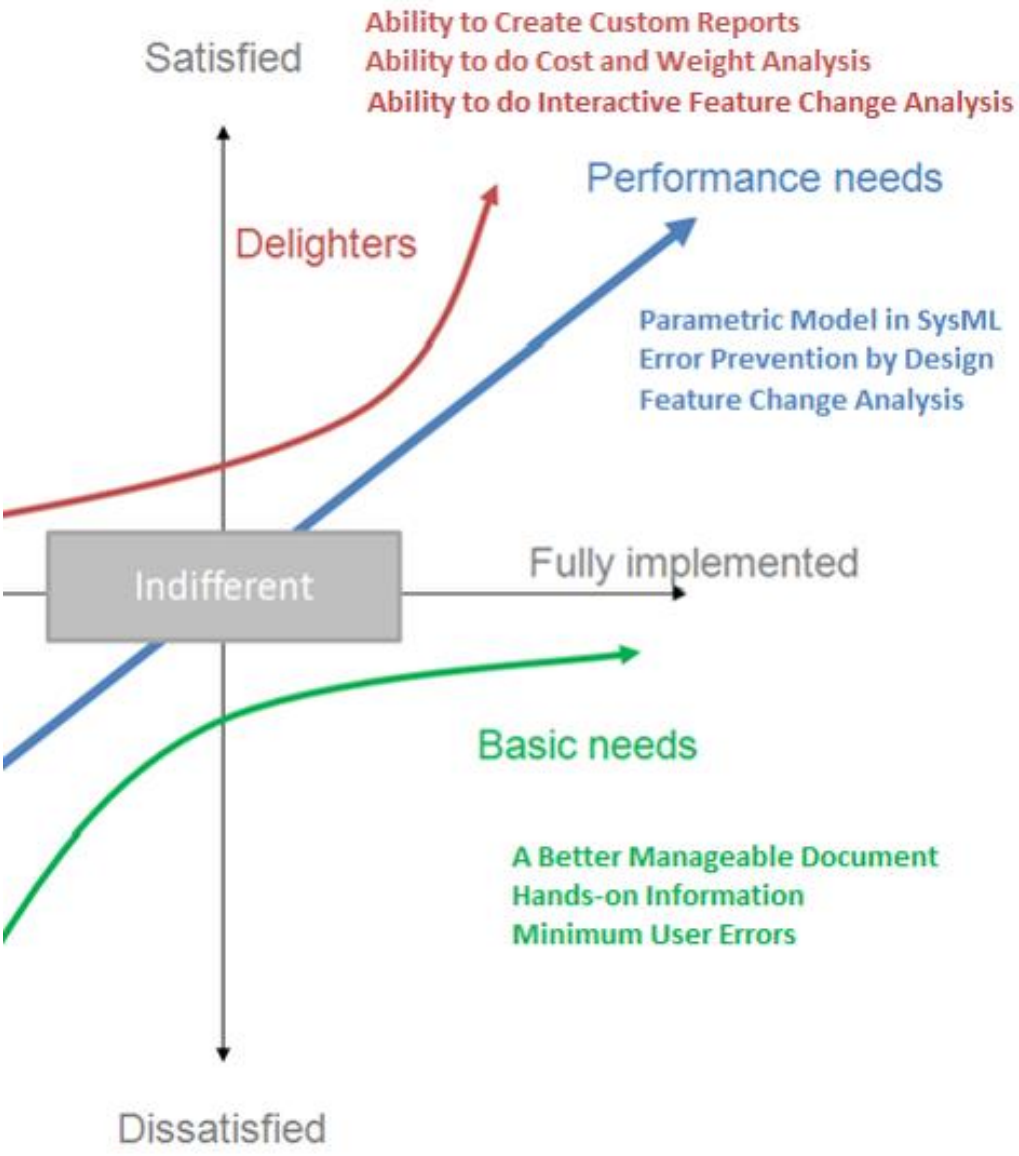


Figure 2-4 Kano Model for Preliminary Requirements

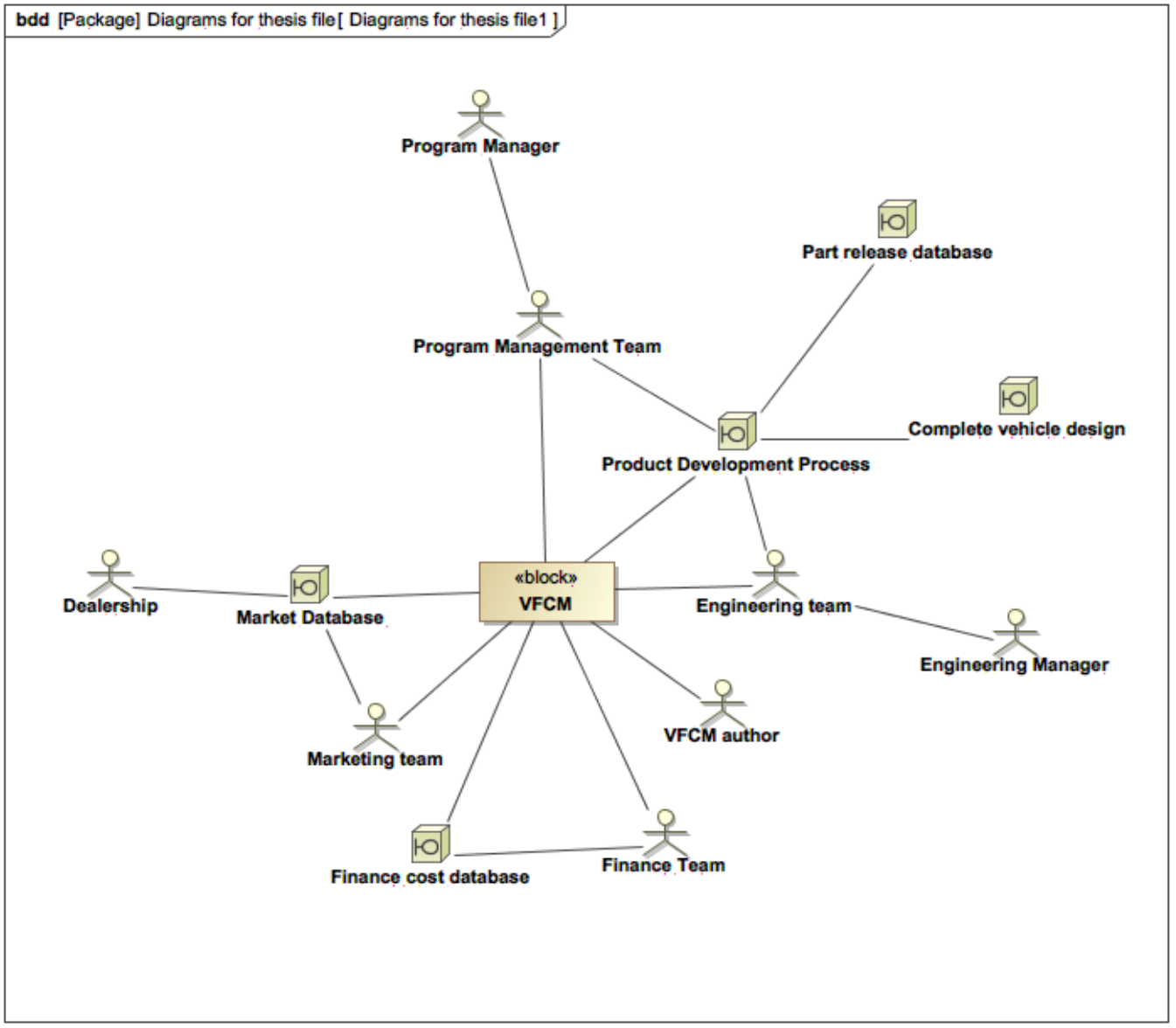


Figure 2-5: VFCM System Context Diagram

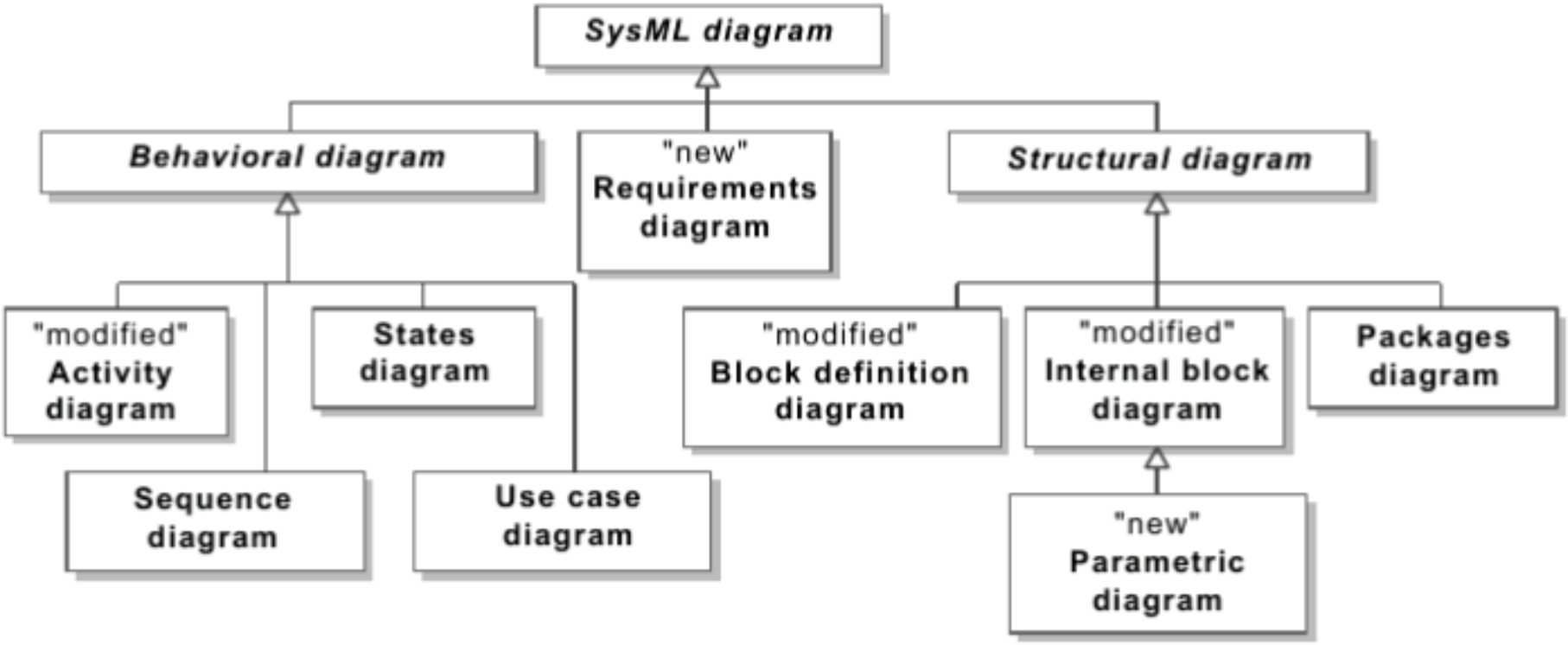


Figure 3-1 Nine Types of SysML Diagrams (Kordon, Hugues, Canals, & Dohet, 2013)

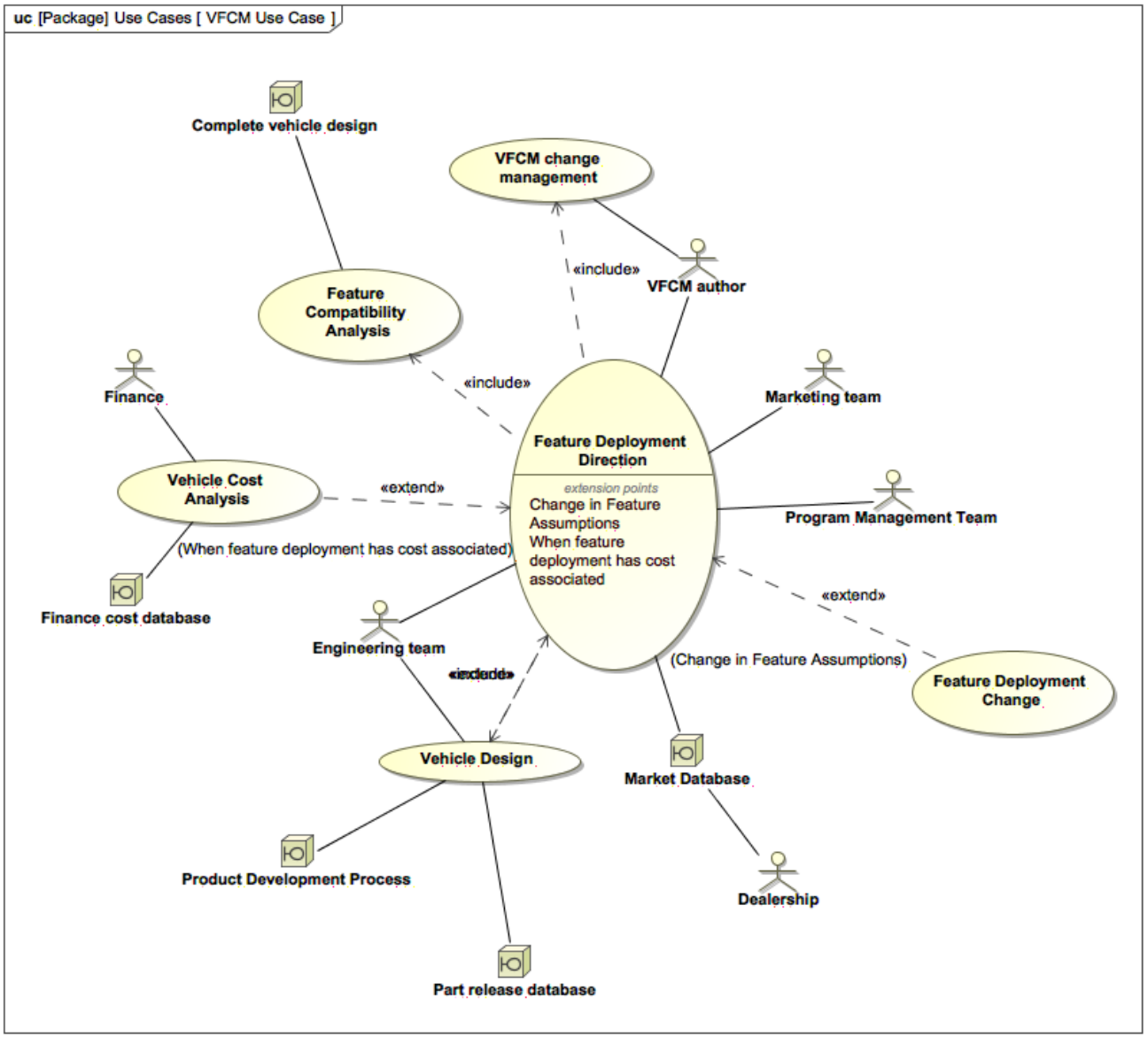


Figure 4-1 VFCM Use Case Diagram

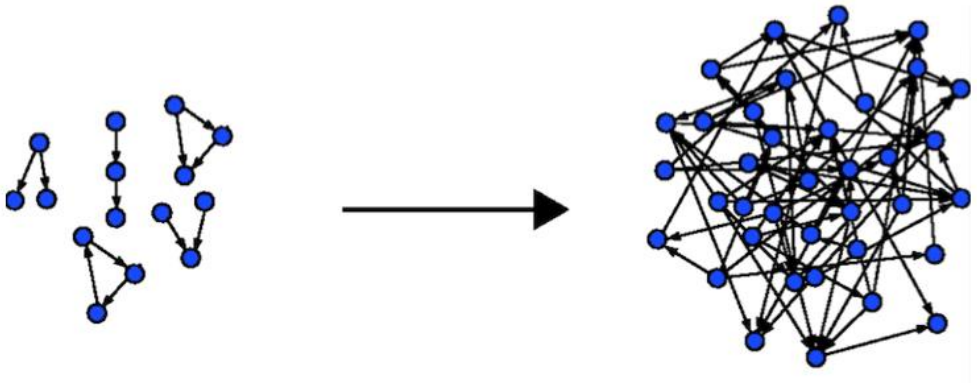


Figure 4-2 Complex Relations of Elements in a Large System

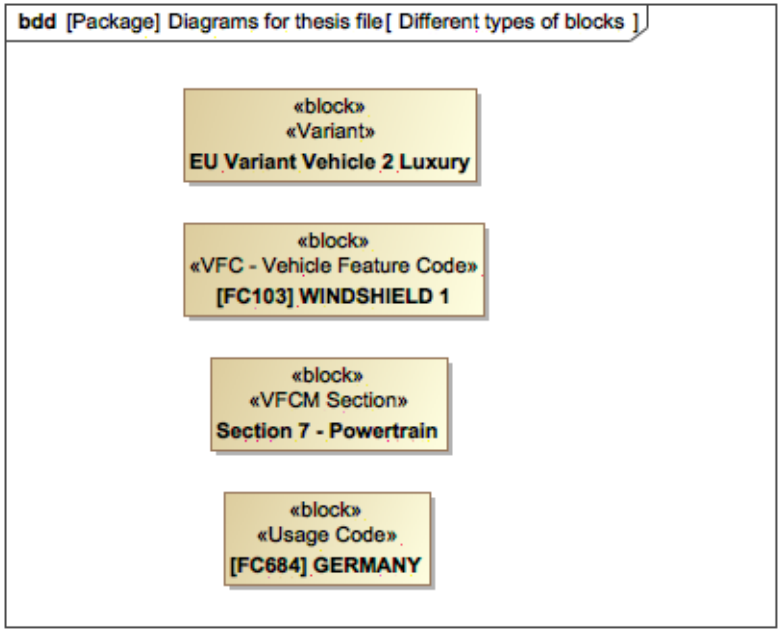


Figure 4-3 Types of Blocks in the VFCM Model (Stereotypes)

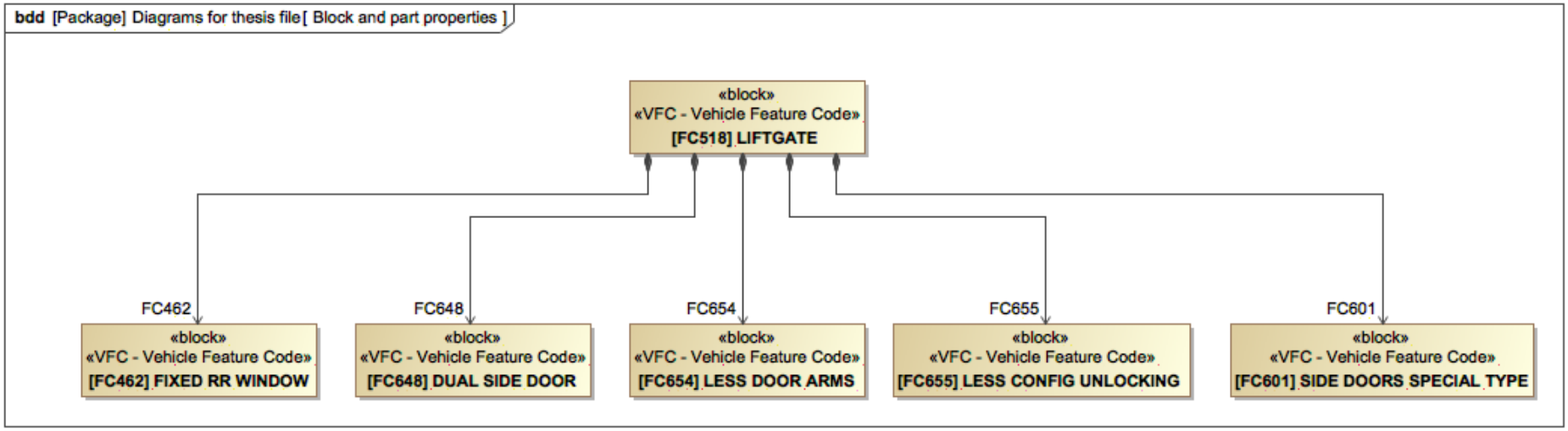


Figure 4-4 Block and Part Properties Model

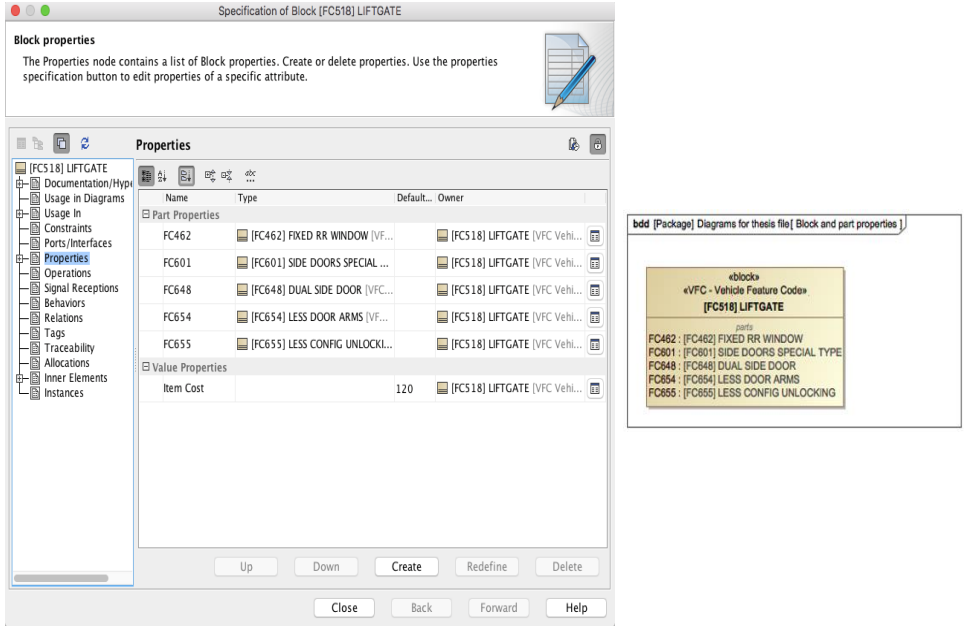


Figure 4-5 Block Specification. Part Properties

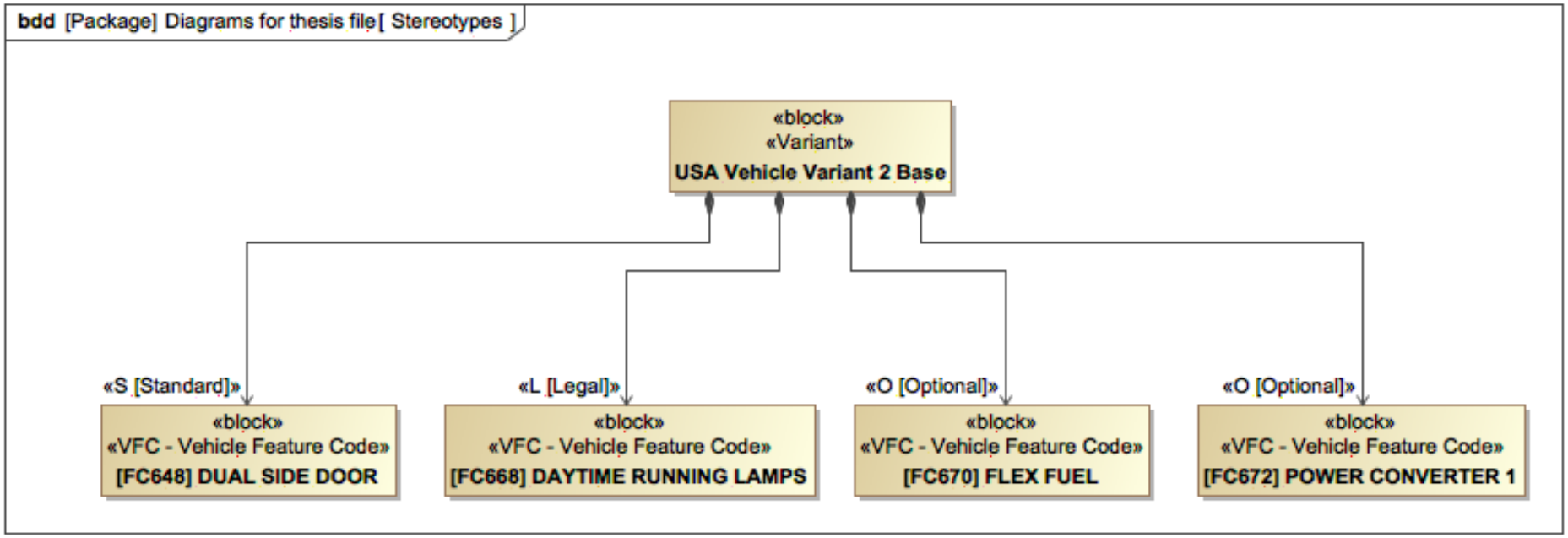
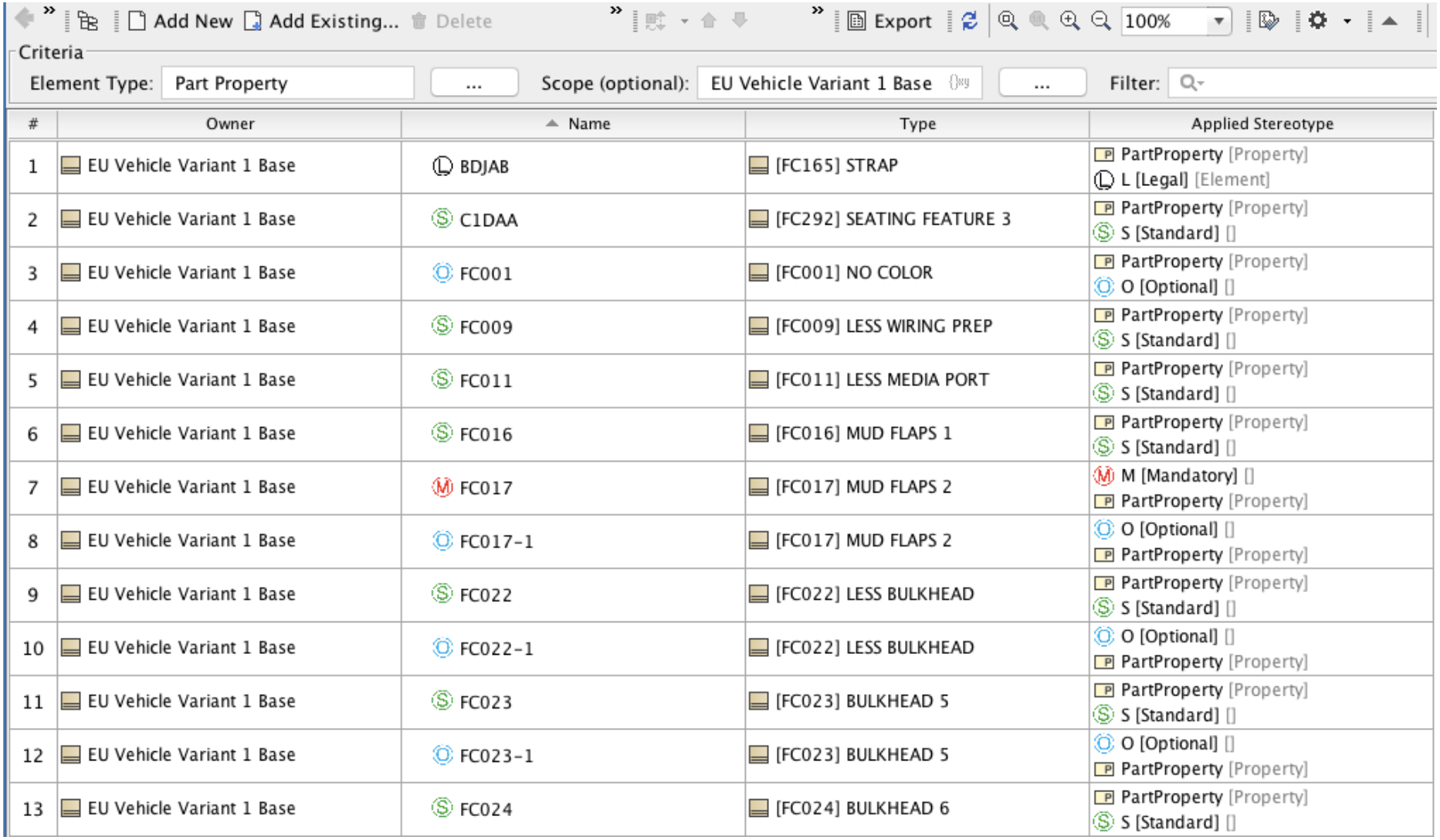


Figure 4-6 Vehicle Variant with associated Features. Example of Stereotypes



#	Owner	Name	Type	Applied Stereotype
1	EU Vehicle Variant 1 Base	BDJAB	[FC165] STRAP	PartProperty [Property] L [Legal] [Element]
2	EU Vehicle Variant 1 Base	C1DAA	[FC292] SEATING FEATURE 3	PartProperty [Property] S [Standard] []
3	EU Vehicle Variant 1 Base	FC001	[FC001] NO COLOR	PartProperty [Property] O [Optional] []
4	EU Vehicle Variant 1 Base	FC009	[FC009] LESS WIRING PREP	PartProperty [Property] S [Standard] []
5	EU Vehicle Variant 1 Base	FC011	[FC011] LESS MEDIA PORT	PartProperty [Property] S [Standard] []
6	EU Vehicle Variant 1 Base	FC016	[FC016] MUD FLAPS 1	PartProperty [Property] S [Standard] []
7	EU Vehicle Variant 1 Base	FC017	[FC017] MUD FLAPS 2	M [Mandatory] [] PartProperty [Property]
8	EU Vehicle Variant 1 Base	FC017-1	[FC017] MUD FLAPS 2	O [Optional] [] PartProperty [Property]
9	EU Vehicle Variant 1 Base	FC022	[FC022] LESS BULKHEAD	PartProperty [Property] S [Standard] []
10	EU Vehicle Variant 1 Base	FC022-1	[FC022] LESS BULKHEAD	O [Optional] [] PartProperty [Property]
11	EU Vehicle Variant 1 Base	FC023	[FC023] BULKHEAD 5	PartProperty [Property] S [Standard] []
12	EU Vehicle Variant 1 Base	FC023-1	[FC023] BULKHEAD 5	O [Optional] [] PartProperty [Property]
13	EU Vehicle Variant 1 Base	FC024	[FC024] BULKHEAD 6	PartProperty [Property] S [Standard] []

Figure 4-7 Vehicle Variant Part Properties Table. Optionality Stereotypes

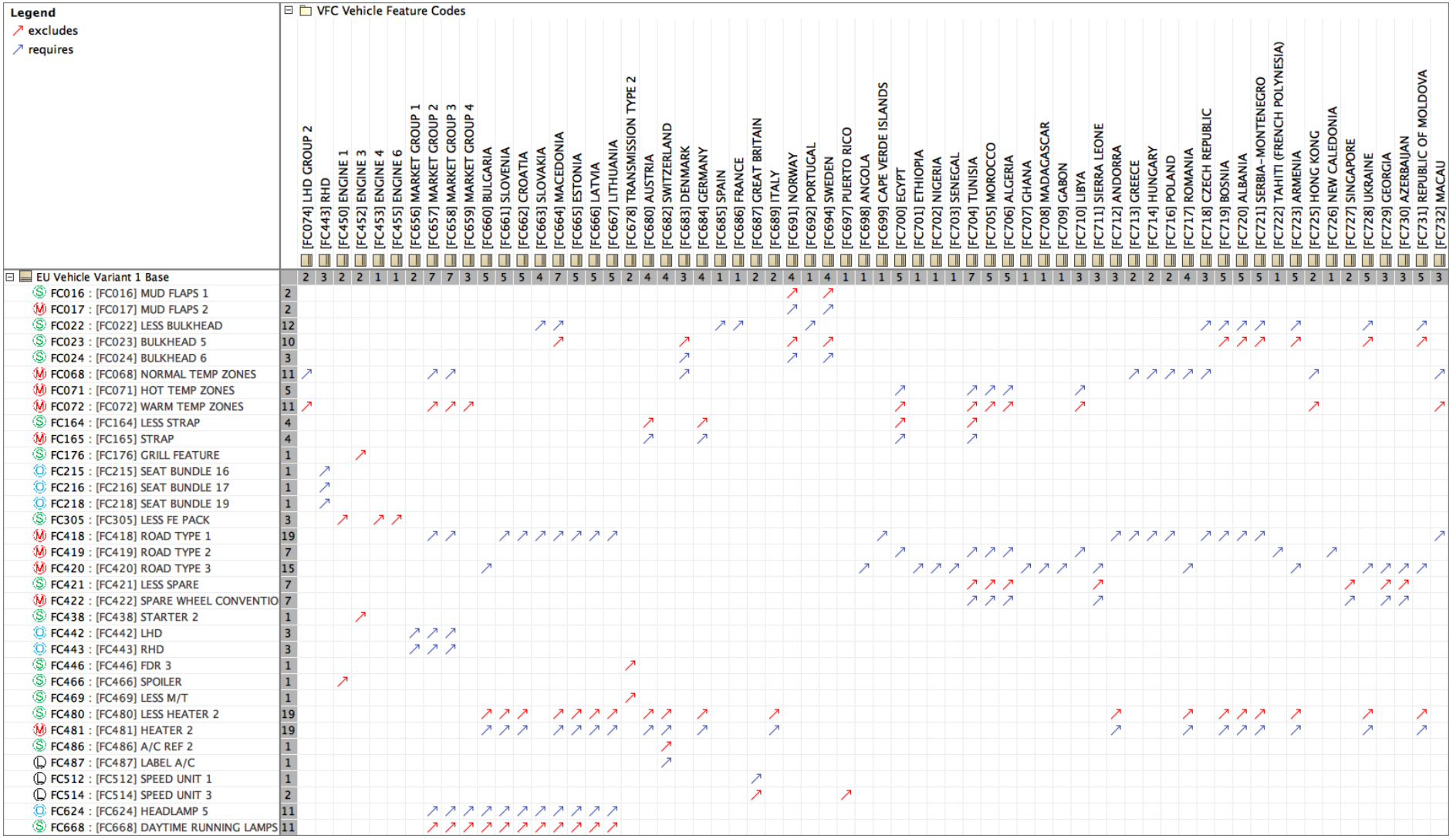


Figure 4-8 Requires / Excludes Relations in SysML Model

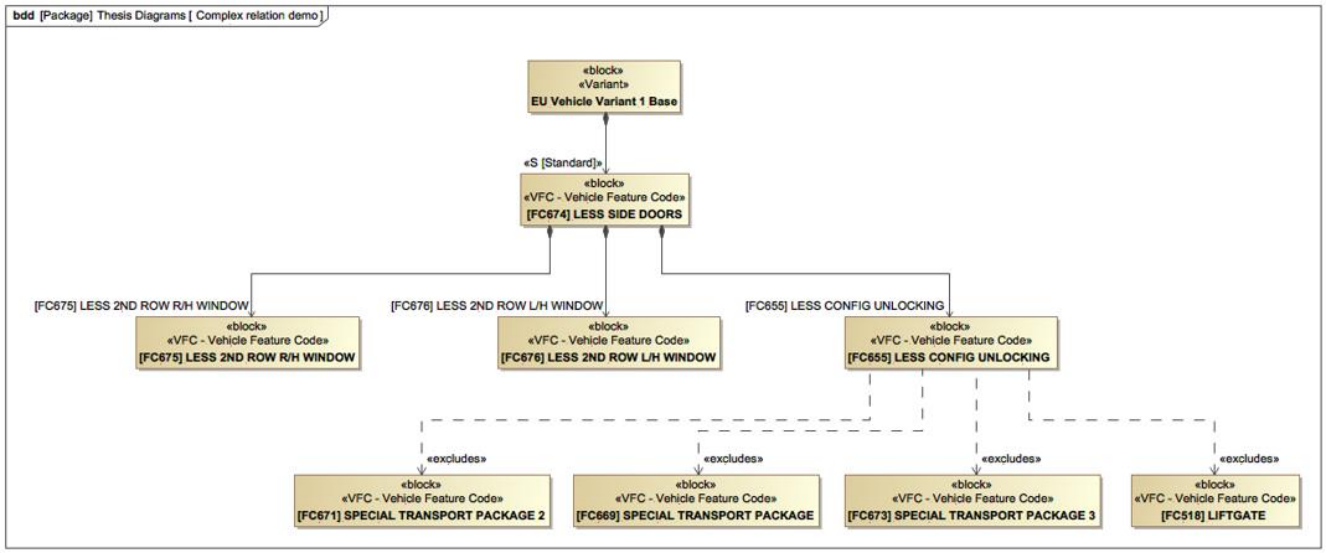


Figure 4-9 Feature Constraints. SysML based VFCM

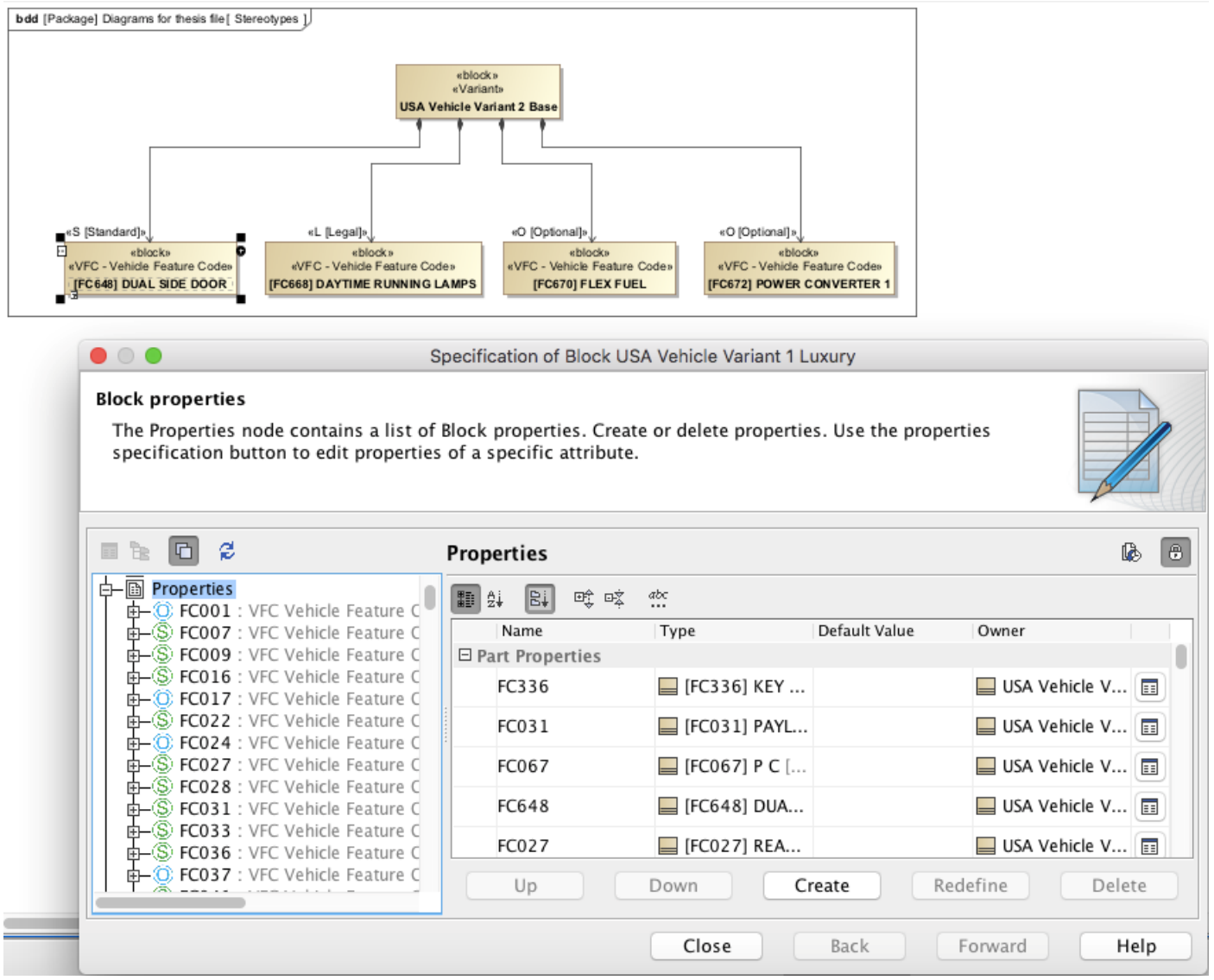


Figure 4-10 Feature Usage Analysis From Block Specification Window

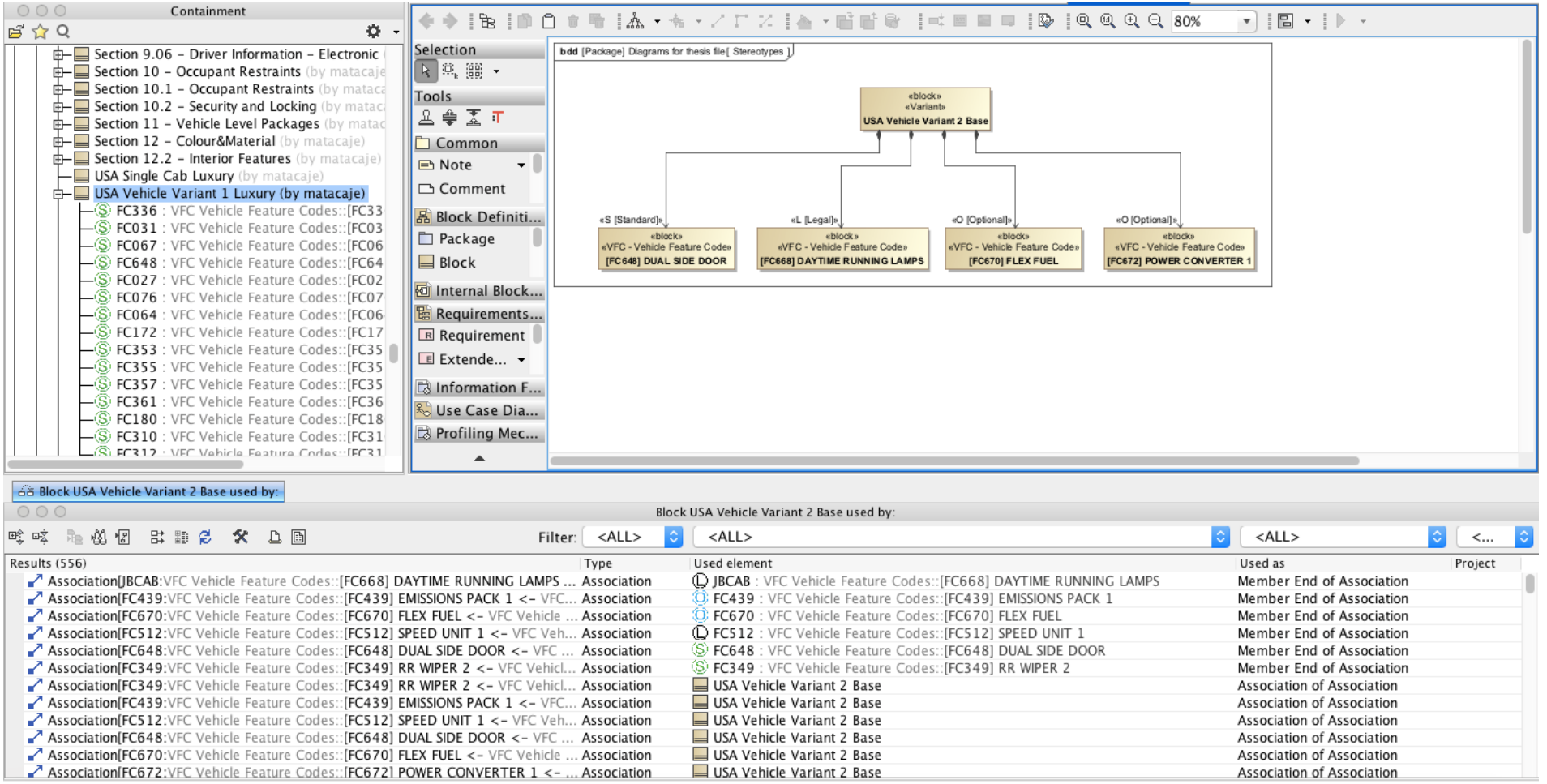


Figure 4-11 Feature Usage Analysis Using the Magic Draw "Used by" Function

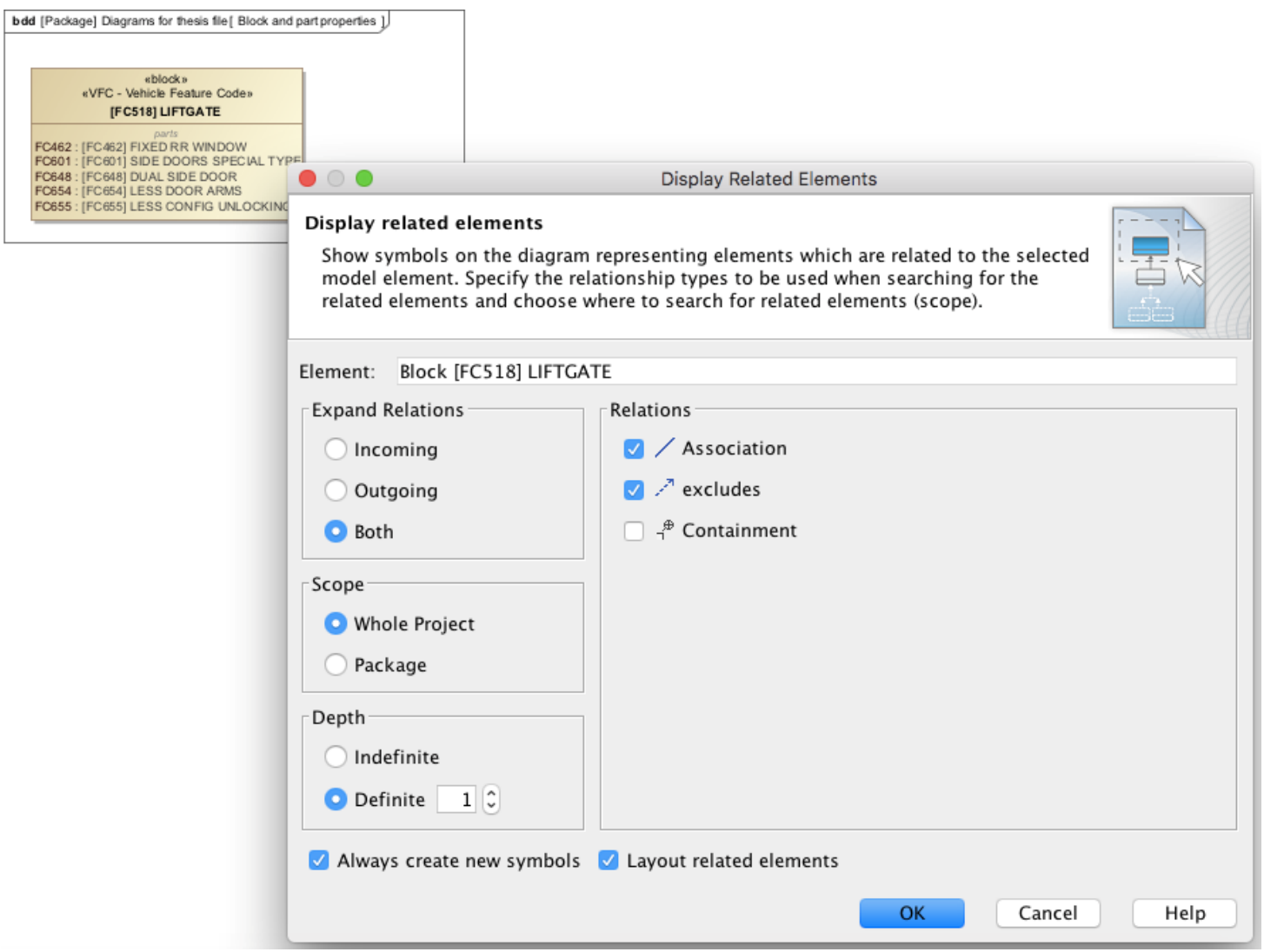


Figure 4-12: Feature Usage Analysis Using "Display Related Elements" Function

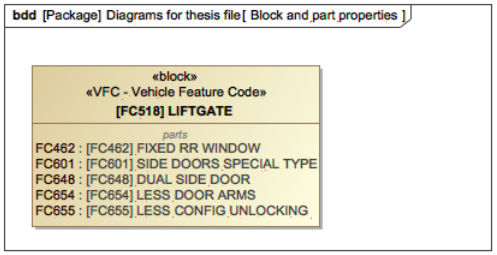


Figure 4-13: Feature Block

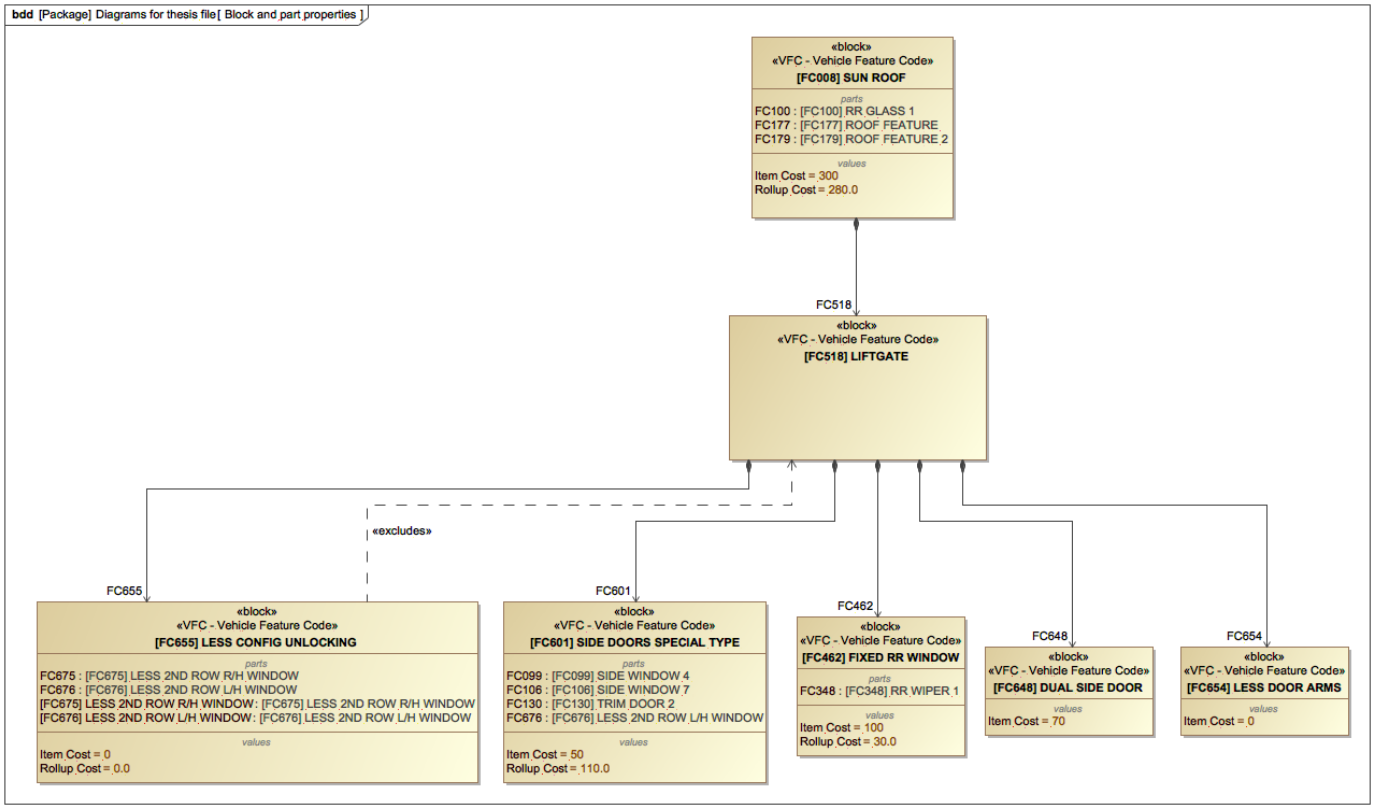


Figure 4-14: Display Related Elements Result. Depth 1-Level

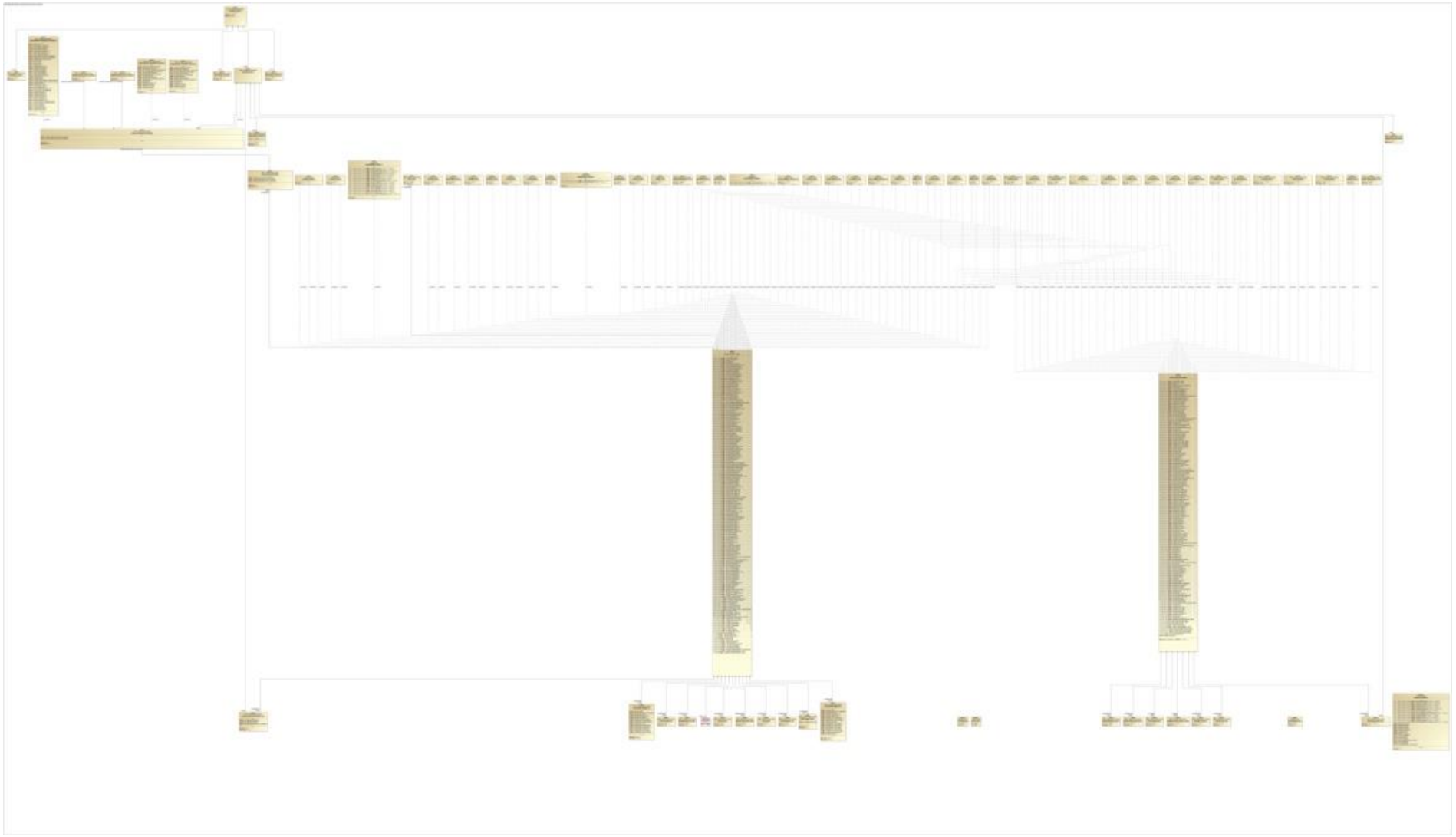


Figure 4-15: Display Related Elements Analysis result. 3 Levels

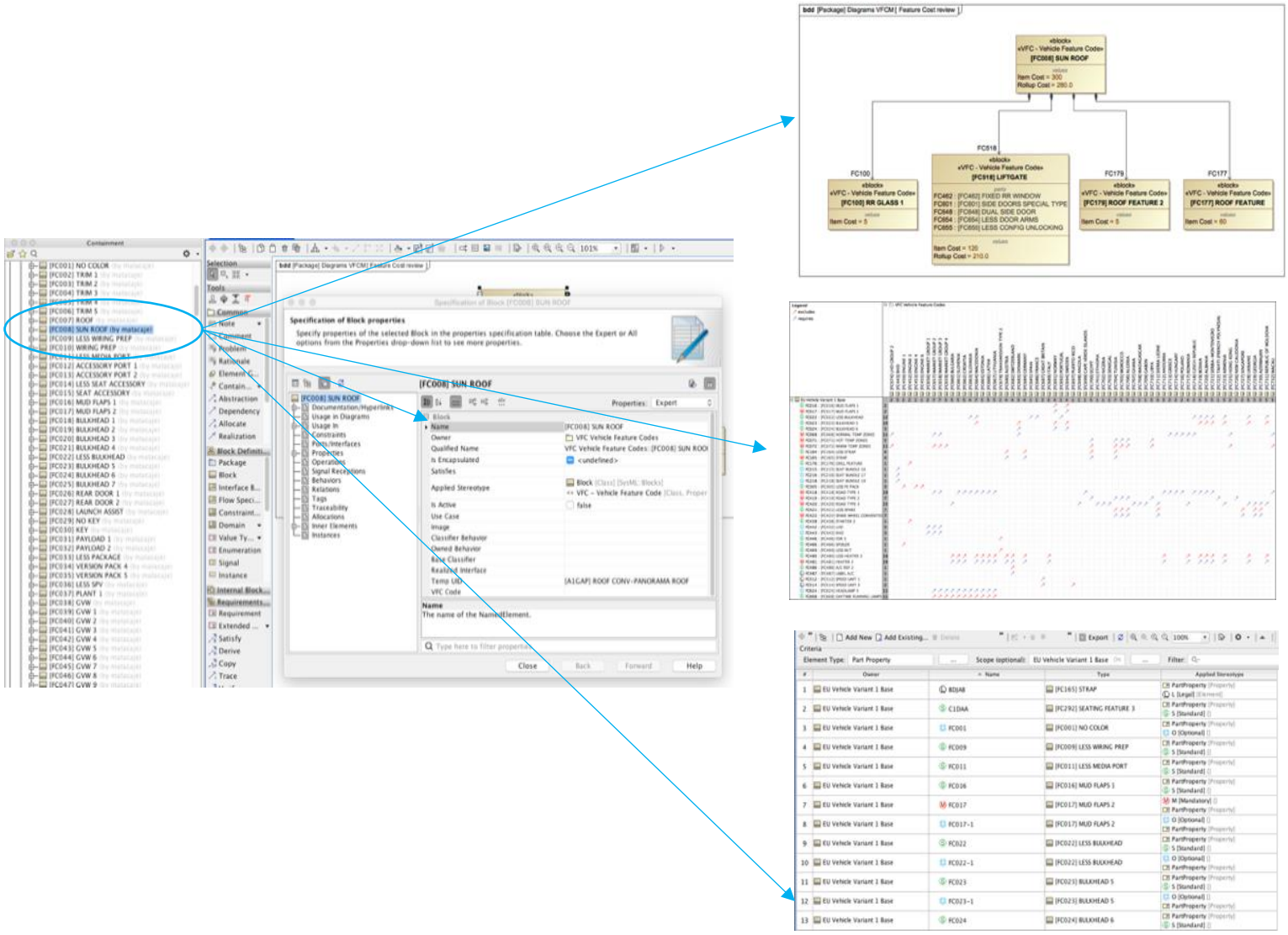
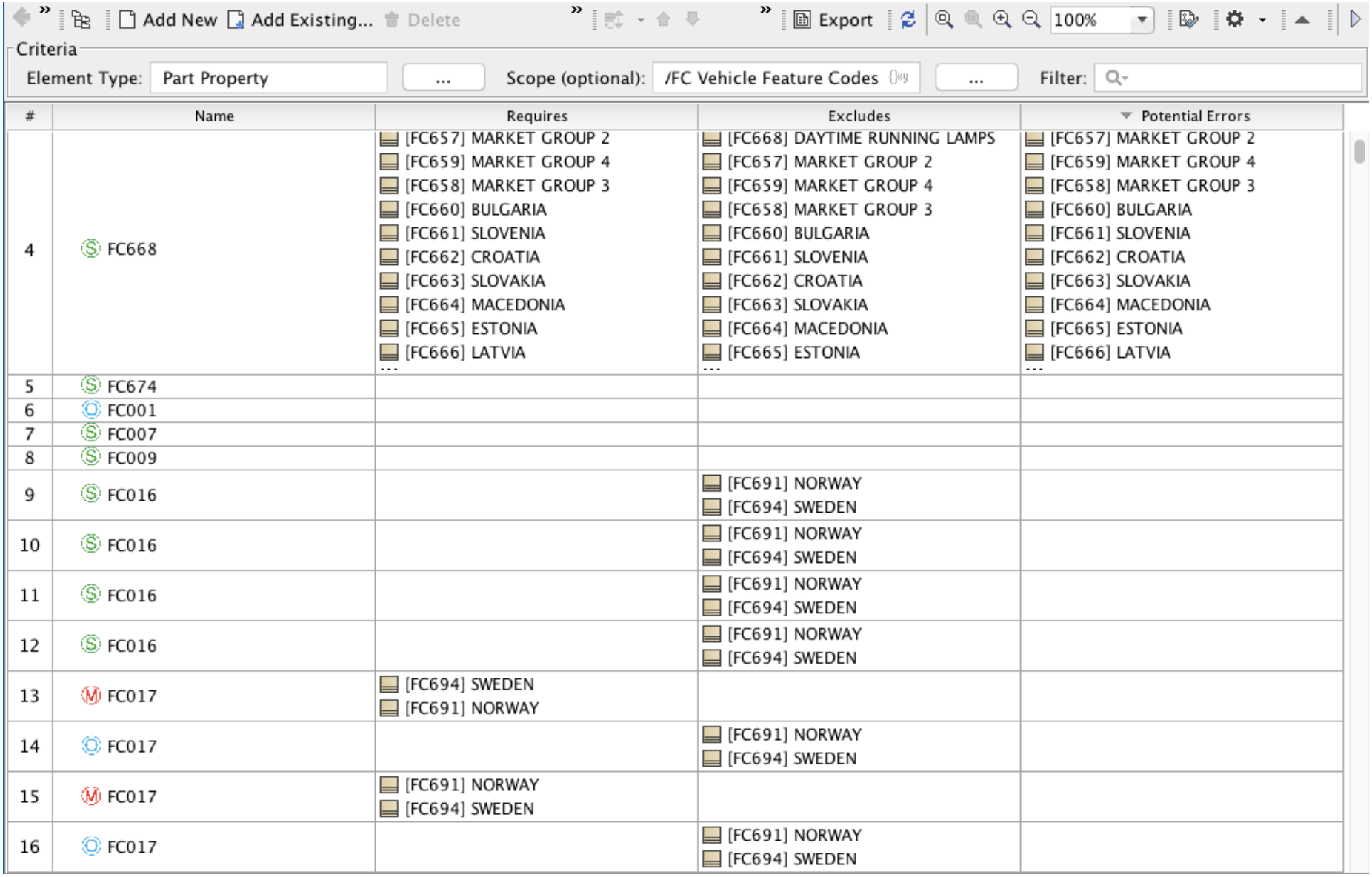


Figure 4-16: Automatic Propagation of the Feature Change in the SysML Model



#	Name	Requires	Excludes	Potential Errors
4	FC668	[FC657] MARKET GROUP 2 [FC659] MARKET GROUP 4 [FC658] MARKET GROUP 3 [FC660] BULGARIA [FC661] SLOVENIA [FC662] CROATIA [FC663] SLOVAKIA [FC664] MACEDONIA [FC665] ESTONIA [FC666] LATVIA ...	[FC668] DAYTIME RUNNING LAMPS [FC657] MARKET GROUP 2 [FC659] MARKET GROUP 4 [FC658] MARKET GROUP 3 [FC660] BULGARIA [FC661] SLOVENIA [FC662] CROATIA [FC663] SLOVAKIA [FC664] MACEDONIA [FC665] ESTONIA ...	[FC657] MARKET GROUP 2 [FC659] MARKET GROUP 4 [FC658] MARKET GROUP 3 [FC660] BULGARIA [FC661] SLOVENIA [FC662] CROATIA [FC663] SLOVAKIA [FC664] MACEDONIA [FC665] ESTONIA [FC666] LATVIA ...
5	FC674			
6	FC001			
7	FC007			
8	FC009			
9	FC016		[FC691] NORWAY [FC694] SWEDEN	
10	FC016		[FC691] NORWAY [FC694] SWEDEN	
11	FC016		[FC691] NORWAY [FC694] SWEDEN	
12	FC016		[FC691] NORWAY [FC694] SWEDEN	
13	FC017	[FC694] SWEDEN [FC691] NORWAY		
14	FC017		[FC691] NORWAY [FC694] SWEDEN	
15	FC017	[FC691] NORWAY [FC694] SWEDEN		
16	FC017		[FC691] NORWAY [FC694] SWEDEN	

Figure 4-17 Error Proofing Tool for Requires / Excludes Relationships

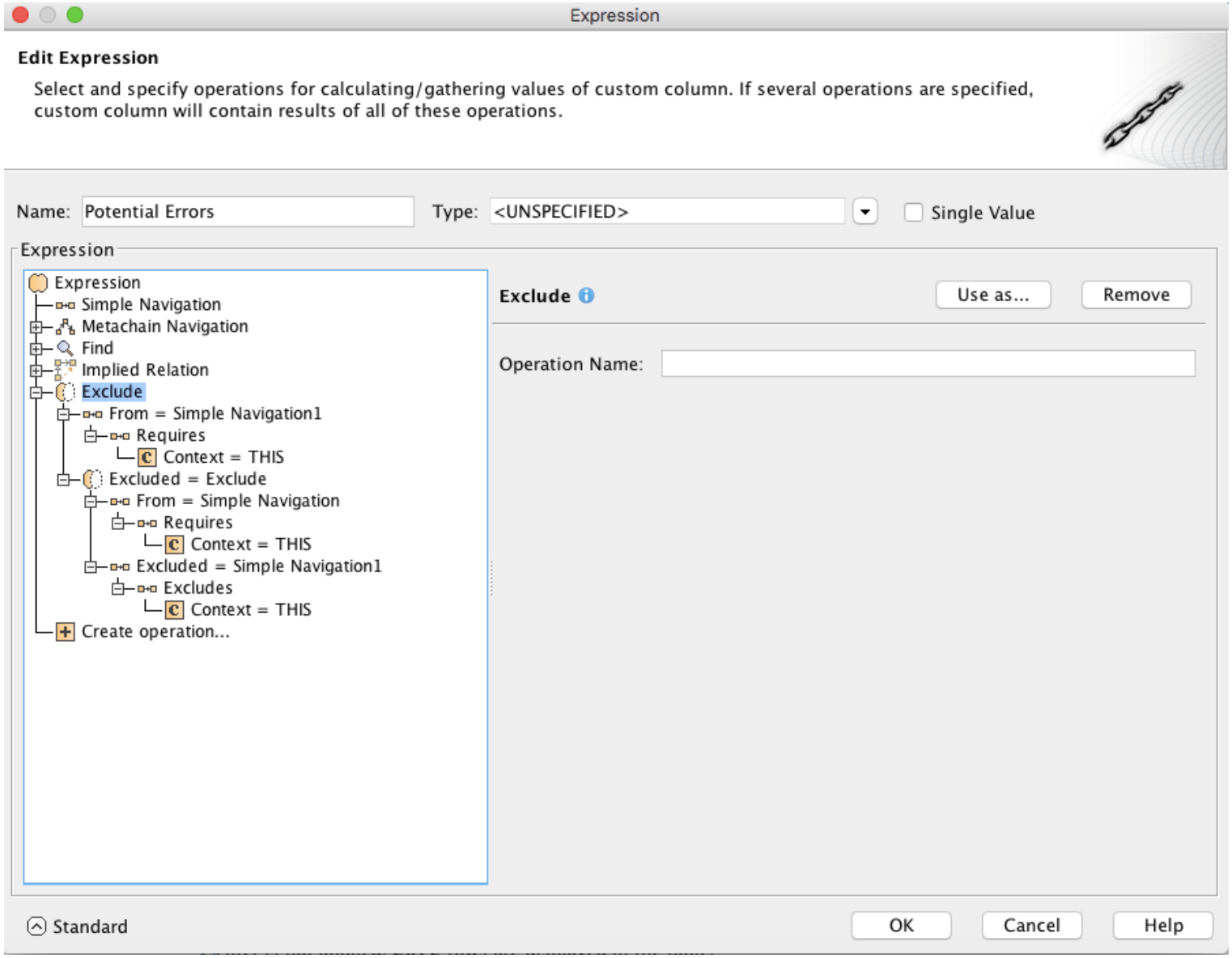


Figure 4-18 Custom Property for Usage in the Error-Proof Table Creation

#	Name	Standard	Optional	Mandatory	Legal	Market	#Stand	#Opti	#Man	#Leg	# Market
1	EU Variant Vehicle 2 Luxury	FC076 : VFC Vehicle Feature Codes::[FC076] LESS MODIFICATION FC376 : VFC Vehicle Feature Codes::[FC376] AIRBAG 8 FC170 : VFC Vehicle Feature Codes::[FC170] LESS LADDER FC018 : VFC Vehicle Feature Codes::[FC018] BULKHEAD 1 FC317 : VFC Vehicle Feature Codes::[FC317] GEAR KNOB 1 FC591 : VFC Vehicle Feature Codes::[FC591] PHONE INTERFACE 1 FC486 : VFC Vehicle Feature Codes::[FC486] A/C REF 2 FC459 : VFC Vehicle Feature Codes::[FC459] BRAKE LIGHT FC019 : VFC Vehicle Feature Codes::[FC019] BULKHEAD 2 FC409 : VFC Vehicle Feature Codes::[FC409] TIRE BRAND ...	FC066 FC037 FC412 FC605 FC616 FC335 FC104 FC021 FC427 FC481 ...	FC418 FC419 G2AAF FC068 FC071 FC165 D17AD FC072 FC017 FC420 ...	FC487 BDJAB : JBCAB : FC514 FC512 	[FC657] MARKET GROUP 2 [FC659] MARKET GROUP 4 [FC658] MARKET GROUP 3 [FC660] BULGARIA [FC661] SLOVENIA [FC662] CROATIA [FC663] SLOVAKIA [FC664] MACEDONIA [FC665] ESTONIA [FC666] LATVIA ...	140	50	11	5	60
2	EU Vehicle Variant 1 Base	FC635 : VFC Vehicle Feature Codes::[FC635] LIGHT FEATURE 4 FC194 : VFC Vehicle Feature Codes::[FC194] MIRROR COLOR 3 FC089 : VFC Vehicle Feature Codes::[FC089] LESS WINDOW FEATURE FC449 : VFC Vehicle Feature Codes::[FC449] EMISSIONS PACK 4 FC617 : VFC Vehicle Feature Codes::[FC617] LESS POD FC355 : VFC Vehicle Feature Codes::[FC355] RR BUMPER 2 FC591 : VFC Vehicle Feature Codes::[FC591] PHONE INTERFACE 1 FC337 : VFC Vehicle Feature Codes::[FC337] KEY FREQ 2 FC457 : VFC Vehicle Feature Codes::[FC457] BRAKE FEATURE 1 FC284 : VFC Vehicle Feature Codes::[FC284] LESS RR ROW 3 ...	FC422- FC195 FC077 FC616 FC442 FC411 FC435 FC650 FC443 FC531 ...	FC072 FC420 FC422 FC481 FC017 FC419 FC165 FC068 FC071 FC418 ...	FC668 FC487 FC514 FC512 BDJAB : 	[FC694] SWEDEN [FC691] NORWAY [FC074] LHD GROUP 2 [FC657] MARKET GROUP 2 [FC658] MARKET GROUP 3 [FC683] DENMARK [FC713] GREECE [FC714] HUNGARY [FC716] POLAND [FC717] ROMANIA ...	155	39	10	5	58
3	USA Vehicle Variant 1 Luxury	FC064 : VFC Vehicle Feature Codes::[FC064] PAINT 1 FC457 : VFC Vehicle Feature Codes::[FC457] BRAKE FEATURE 1 FC093 : VFC Vehicle Feature Codes::[FC093] WINDOW FEATURE 4 FC306 : VFC Vehicle Feature Codes::[FC306] FE PACK 1 FC642 : VFC Vehicle Feature Codes::[FC642] STOP LAMP FC126 : VFC Vehicle Feature Codes::[FC126] LESS SEAT FEATURE FC270 : VFC Vehicle Feature Codes::[FC270] LESS STOWAGE FC444 : VFC Vehicle Feature Codes::[FC444] FDR 1 FC466 : VFC Vehicle Feature Codes::[FC466] SPOILER FC160 : VFC Vehicle Feature Codes::[FC160] LESS LOAD FLOOR ...	FC177 FC668 FC057 FC439 FC679 FC431 FC548 FC179 FC024 FC672 ...	FC072 FC068 FC418 	FC512 FC458 FC487 	[FC455] ENGINE 6 [FC678] TRANSMISSION TYPE 2 [FC452] ENGINE 3 [FC679] TRANSMISSION TYPE 3 [FC453] ENGINE 4 [FC695] USA [FC697] PUERTO RICO 	148	45	3	3	7
4	USA Vehicle Variant 2 Base	FC476 : VFC Vehicle Feature Codes::[FC476] DUCT FC530 : VFC Vehicle Feature Codes::[FC530] FEATURE X 3 FC603 : VFC Vehicle Feature Codes::[FC603] POWER OUTLET 2 FC311 : VFC Vehicle Feature Codes::[FC311] DOOR HANDLE 3 FC376 : VFC Vehicle Feature Codes::[FC376] AIRBAG 8 FC626 : VFC Vehicle Feature Codes::[FC626] HEADLAMP 7 FC051 : VFC Vehicle Feature Codes::[FC051] LESS UPGRADE FEATURI FC164 : VFC Vehicle Feature Codes::[FC164] LESS STRAP FC463 : VFC Vehicle Feature Codes::[FC463] TRACKING 1 FC181 : VFC Vehicle Feature Codes::[FC181] WIDE BODYSIDE MLDG- ...	FC464 FC082 FC650 FC653 FC455 FC001 FC080 FC670 FC592 FC439 ...	FC072 FC068 FC070 	FC513 FC512 JBCAB : 	[FC455] ENGINE 6 [FC678] TRANSMISSION TYPE 2 [FC452] ENGINE 3 [FC679] TRANSMISSION TYPE 3 [FC453] ENGINE 4 [FC697] PUERTO RICO [FC695] USA [FC696] CANADA	130	41	3	3	8

Figure 4-19: Vehicle Variants Feature and Markets Summary Table

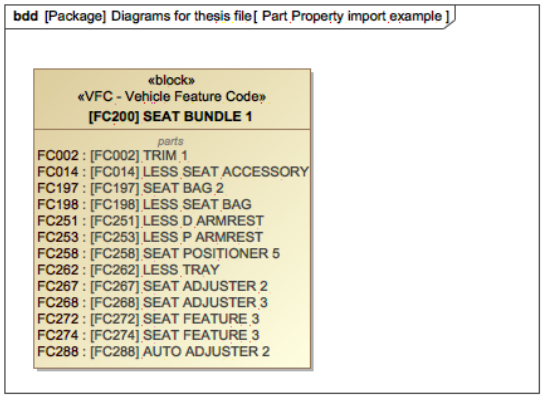


Figure 4-20: Part Properties imported using Magic Draw .CSV import tool

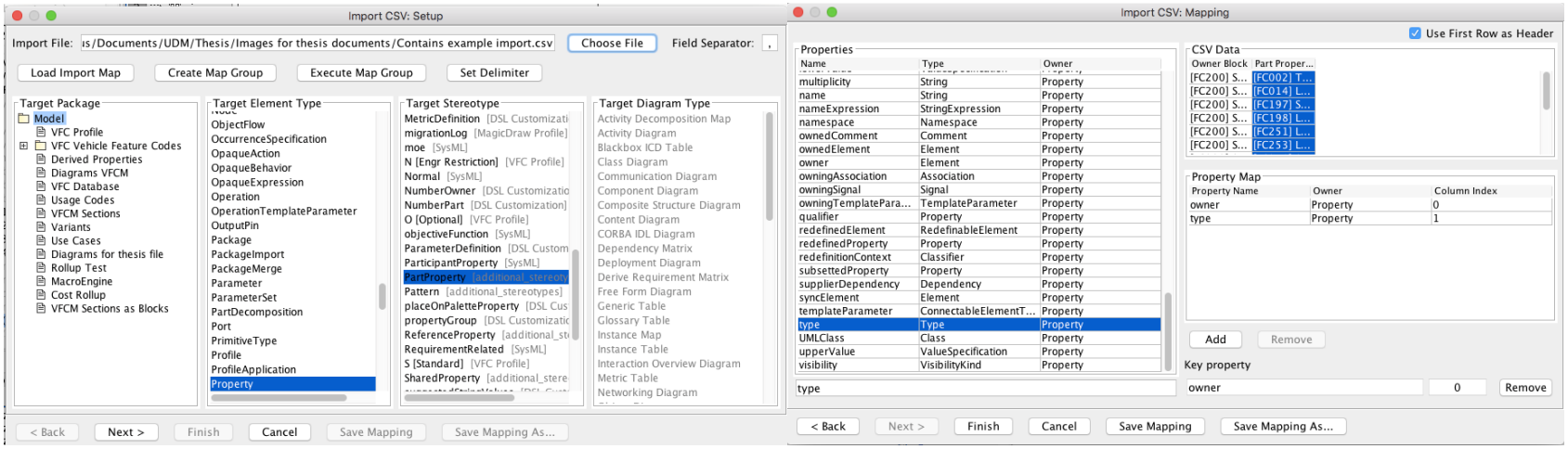


Figure 4-21: Magic Draw "Import from .CSV" Tool Interface

Figure 4-22: Magic Draw "Import from .CSV" Column Selection Screen

#	Owner	Default Value	Feature Cost [Dollars]
70	[FC041] GVW 3	0	0
71	[FC440] EMISSIONS PACK 2	600	600
72	[FC632] LIGHT FEATURE 1	5	5
73	[FC219] SEAT BUNDLE 20	220	220
74	[FC084] REPAIR KIT	5	5
75	[FC166] LESS GRAB HANDLE	0	0
76	[FC451] ENGINE 2	3000	3000
77	[FC125] CUP HOLDER	8	8
78	[FC149] LESS HOOK	0	0
79	[FC535] FEATURE X 8	100	100
80	[FC343] DOOR LOCKS 5	10	10
81	[FC248] SEAT BUNDLE 49	300	300
82	[FC146] DOOR HANDLE 2	20	20
83	[FC003] TRIM 2	5	5
84	[FC417] LUG NUT 2	5	5
85	[FC284] LESS RR ROW 3	0	0
86	[FC523] SCREEN 1	0	0
87	[FC053] UPGRADE FEATURE P...	250	250
88	[FC117] GLOVE BOX 1	0	0
89	[FC283] RR ROW 2	10.0	10.0
90	[FC202] SEAT BUNDLE 3	320	320
91	[FC445] FDR 2	5	5
92	[FC311] DOOR HANDLE 3	5	5
93	[FC273] LESS SEAT FEATURE 3	0	0
94	[FC470] M/T	2000	2000
95	[FC193] MIRROR COLOR 2	5	5
96	[FC446] FDR 3	5	5
97	[FC352] JETS	20	20

Figure 5-1: Features Block Value Properties (Sample Cost Database)

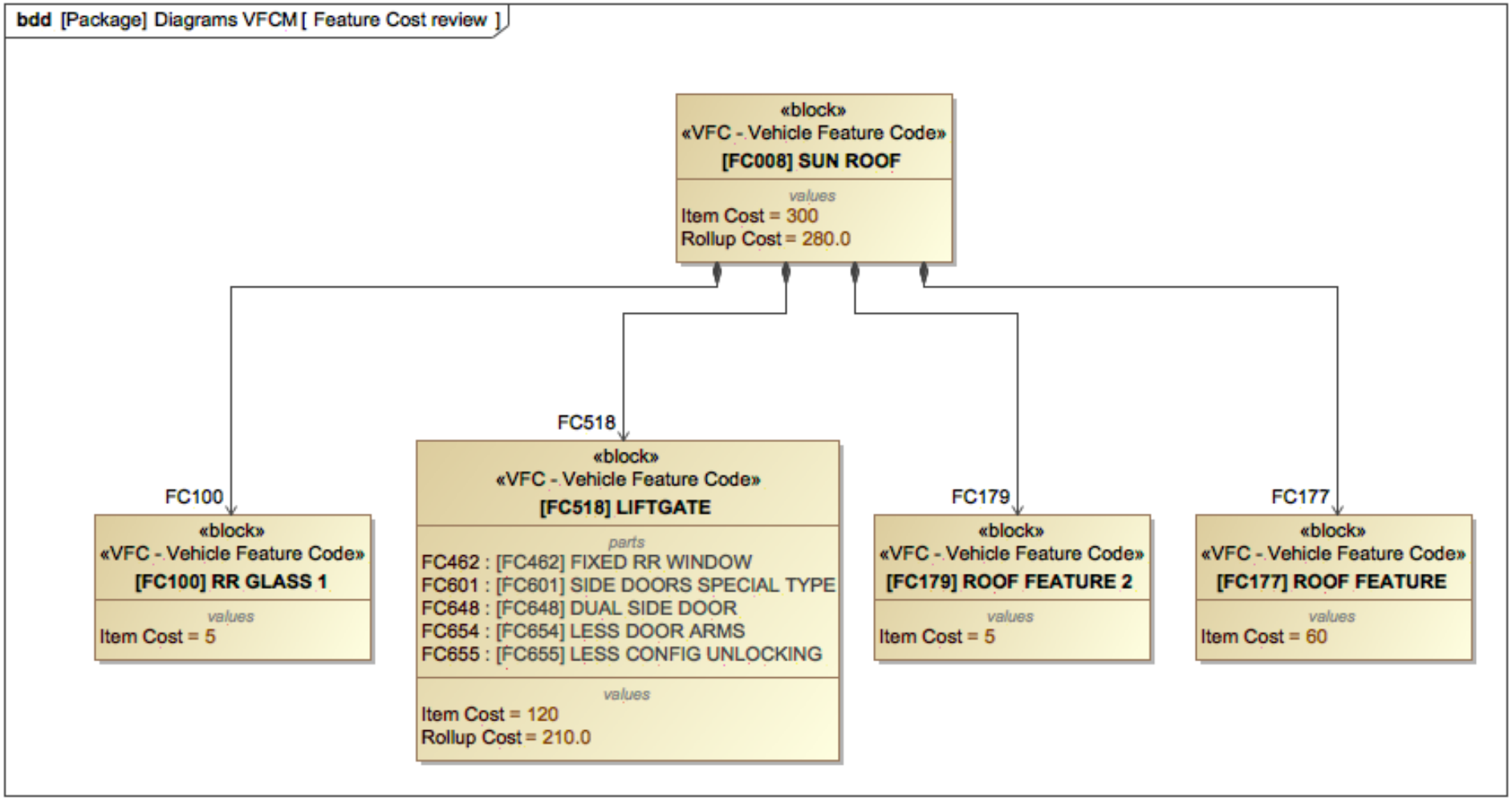


Figure 5-2: BDD for Feature Cost and Cost Roll-up Analysis

#	Owner	Default Value	Type
1	USA Vehicle Variant 2 Base	16570.0	cost [dollars]
2	USA Vehicle Variant 1 Luxury	19766.0	cost [dollars]
3	EU Variant Vehicle 2 Luxury	19598.0	cost [dollars]
4	EU Vehicle Variant 1 Base	10186.0	cost [dollars]

Figure 5-3: Cost Rollup Analysis Result for Complete Vehicle Variants

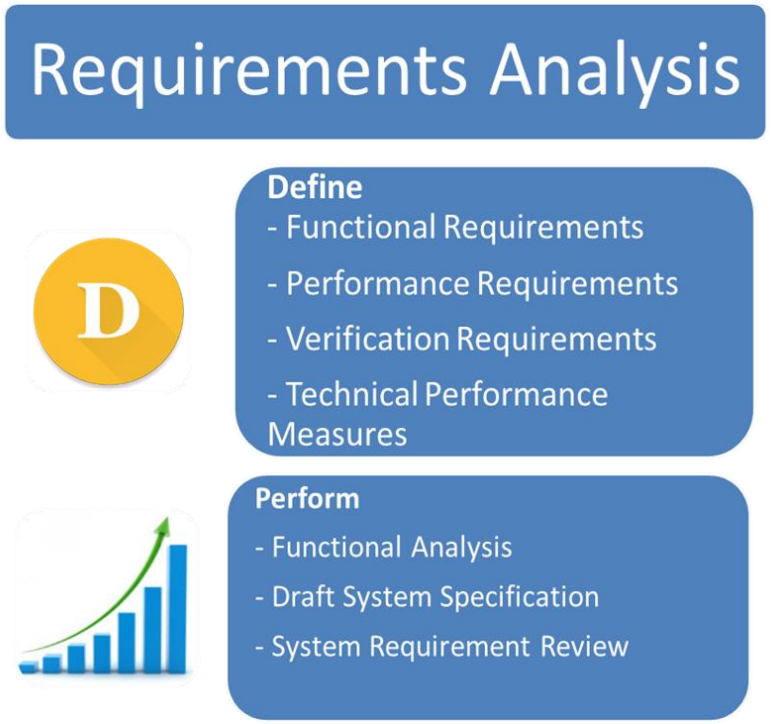


Figure 6-1 Next Steps. Requirement Analysis

Software Package	Creator / Publisher	License
Agilan	Visual Paradigm	Commercial
Artisan Studio	Atego	Commercial
Enterprise Architect	Sparx Systems	Commercial
Cameo Systems Modeler /	No Magic	Commercial
Rhapsody	IBM Rational	Commercial
UModel	Altova	Commercial
Modelio	Modeliosoft	Open Source

Papyrus	Atos Origin	Open Source
SysML Solution	Concept Draw	Commercial
Lattix Architect	Lattix	Commercial
Software Ideas Modeler	Dusan Rodina	Open Source
SysML Designer	ObeoNetwork	Open Source
SCADE System	Esterel Technologies	Commercial

Table 3-1 Software Tools for Systems Modeling (Kraus, Papaioannou, & Sivas, 2015)

	Vehicle Variants			
Feature List	USA Single Cab Base	USA Double Cab Luxury	…	South America Single Cab Base
Section. Suspension				
Tire 1				
Tire 2				
…				
Section. Powertrain				
Engine 1				
Engine 2				
…				

Table 4-1 VFCM Features & Options Deployment Matrix (Example)

	Feature	Feature Code	**Vehicle Variant 1**
	FR WIPER	FC350	
Relations with other features	Contains RR MIRROR 1 (FC184) when {BULKHEAD 2 (FC018), BULKHEAD 3 (FC020), BULKHEAD 5 (FC023)} is not present, and when FIXED RR WINDOW (FC462) is present, and when CAMERA 2 (FC612) is not present		Contains
	Feature	Feature Code	**Vehicle Variant 1**
	RR MIRROR	FC184	
Relations with other features	Included in FR WIPER (FC350) when {BULKHEAD 2 (FC018), BULKHEAD 3 (FC020), BULKHEAD 5 (FC023)} is not present, and when FIXED RR WINDOW (FC462) is present, and when CAMERA 2 (FC612) is not present		Includes

Table 4-2: Example of Feature Relationships

Owner	Feature Code
LIFTGATE Contains SIDE DOORS SPECIAL TYPES (FC601) Contains available DUAL SIDE DOOR (FC648) Contains LESS DOOR ARMS (FC654) Contains FIXED RR WINDOW (FC462) Contains LESS CONFIG UNLOCKING (FC655)	FC518
Parts	**Feature Code**
SIDE DOORS SPECIAL TYPES Included in LIFTGATE (FC518)	FC601
DUAL SIDE DOOR Included in LIFTGATE (FC518)	FC648
LESS DOOR ARMS Included in LIFTGATE (FC518)	FC654
FIXED RR WINDOW Included in LIFTGATE (FC518)	FC462
LESS CONFIG UNLOCKING Included in LIFTGATE (FC518)	FC655

Table 4-3 VFCM Contains/Includes Relationships (Extract From Different Parts of Feature and Options Matrix)

Feature	Feature Code	Variant 1
RHD Requires {MARKET GROUP 1 (FC656), MARKET GROUP 2 (FC657), MARKET GROUP 3 (FC658)}	FC443	O*/- O[1]
HEADLAMP 5 Excludes {MARKET GROUP 2 (FC657), MARKET GROUP 4 (FC659), MARKET GROUP 3(FC658), BULGARIA (FC660), SLOVENIA (FC661), CROATIA (FC662), SLOVAKIA (FC663), MACEDONIA (FC664), ESTONIA (FC665), LATVIA (FC666), LITHUANIA (FC667)}	FC624	O*/- O[1]

Table 4-4 Requires/Excludes Relations in “Document-based” VFCM

Feature	Feature Code	Variant 1
LESS SIDE DOORS	FC674	S*
Contains LESS 2ND ROW R/H WINDOW (FC675)		C[1]
Contains LESS 2ND ROW L/H WINDOW (FC676)		C[2]
Contains LESS CONFIG UNLOCKING (FC655) when LIFTGATE (FC518) is not present, and when {SPECIAL TRANSPORT PACKAGE (FC669), SPECIAL TRANSPORT PACKAGE 2 (FC671), SPECIAL TRANSPORT PACKAGE 3 (FC673)} is not present		C[3]

Table 4-5 Feature Constraints. “Document-based” **VFCM**

Owner Block	Part Property Block
[FC200] SEAT BUNDLE 1	[FC002] TRIM 1
[FC200] SEAT BUNDLE 1	[FC014] LESS SEAT ACCESSORY
[FC200] SEAT BUNDLE 1	[FC197] SEAT BAG 2
[FC200] SEAT BUNDLE 1	[FC198] LESS SEAT BAG
[FC200] SEAT BUNDLE 1	[FC251] LESS D ARMREST
[FC200] SEAT BUNDLE 1	[FC253] LESS P ARMREST
[FC200] SEAT BUNDLE 1	[FC258] SEAT POSITIONER 5
[FC200] SEAT BUNDLE 1	[FC262] LESS TRAY
[FC200] SEAT BUNDLE 1	[FC267] SEAT ADJUSTER 2
[FC200] SEAT BUNDLE 1	[FC268] SEAT ADJUSTER 3
[FC200] SEAT BUNDLE 1	[FC272] SEAT FEATURE 3
[FC200] SEAT BUNDLE 1	[FC274] SEAT FEATURE 3
[FC200] SEAT BUNDLE 1	[FC288] AUTO ADJUSTER 2

Table 4-6: Sample .CSV table to import Feature Block Part Properties