2007

An evaluation of OMG SysML 1.0a standard conformance between modelling tools

Andrew James Campbell

Edith Cowan University

Copyright Warning

"An Evaluation of OMG SysML 1.0a Standard Conformance between Modelling Tools"

A dissertation submitted in partial fulfilment of the requirements for the degree of

Bachelor of Science (Computer Science) – Honours

Author: Andrew James Campbell

Student Number: 6042643

Faculty of Computing, Health and Science

Edith Cowan University

Supervisor: Mr. M Johnstone

Submission Date: October 31st, 2007

USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

Abstract

The SysML is a recent introduction to modelling languages for the systems engineering domain. Modelling tools are offering support for its notation. Studies related to the UML have indicated that modelling tools lack compliance to the UML language. This issue may apply equally to the SysML and the aim of this research is to investigate that language compliance issue.

The first phase of this research is concerned with the compliance of current modelling tools to the SysML 1.0a Final Adopted Specification (FAS). It consists of a comparative evaluation of candidate tools based on an ideal framework derived from the language specification. The second research phase consists of an interpretive evaluation. It is concerned with the ability of SysML modelling tools to consistently represent a modelling problem and this problem is derived from the language specification.

This research may benefit future studies in the field of modelling tool evaluations, particularly studies on the effects of modelling tools with varying compliance to the SysML specification.

Copyright and Access Declaration

I certify that this thesis does not, to the best of my knowledge and belief:

(i) incorporate without acknowledgment any material previously submitted for a degree or diploma in any institution of higher education;

(ii) contain any material previously published or written by another person except where due reference is made in the text of this thesis; or

(iii) contain any defamatory material;

Signed: .....

Dated: .....

Contents

Tables and Figures

Acknowledgements

"I wish to dedicate this thesis to my father; my family in Australia, Scotland and around the world; and my friends. I appreciate their support and understanding throughout my studies."

"I wish to thank my supervisor, Mike. I am grateful for his tremendous assistance during my course of research and for his clarity and dedication."

"I wish to thank Dr Darren Kelly and Dr Nick Hauser (Bragg Institute of ANSTO, Lucas Heights Research Laboratories, N.S.W) for their inspiration and advice."

(Andrew James Campbell, 2007)

1 Introduction

This section will introduce disciplines and concepts related to this research. An introduction to the aims and implications of this research will be presented, along with the questions that were used to guide this research.

1.1 Background

Models are necessary simplifications of reality. The process of building models is called "modelling" (Kühne, 2006). Modelling is typically used in software engineering to define ideal representations of software under development. "Object Modelling" is a technique from this discipline which uses conceptual objects to define models of reality. Visual modelling languages have been developed to accomplish modelling in specific domains and these are termed "domain-specific" modelling languages. One such language is the Systems Modelling Language (SysML), which has emerged to assist "system modelling" within systems engineering. It incorporates software engineering concepts from its language foundation, the Unified Modelling Language (UML).

Computer-aided Software Engineering (CASE) tools or modelling tools have been developed to accomplish modelling using the UML and its extensions. This research is concerned with tools for the SysML and their relationship with a specific version of the language's specification. The following sections will provide background to this research.

1.1.1 Systems Engineering

The engineering practitioners of the International Council on Systems Engineering (INCOSE) use the following official definition for "systems engineering" (INCOSE, 2006, pp. 2):

> "Systems Engineering is an engineering discipline whose responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are satisfied in a high quality, trustworthy, cost efficient and schedule compliant manner throughout a system's entire life cycle."

The practitioners within this field are called System Engineers and have responsibilities that focus on designing systems that are aligned with a customer's needs (Burks, 1991). Requirements definition, analysis and confirmation, along with system design, functional definition and analysis, are amongst the major activities within systems engineering (Bahill & Dean, 2007; Kayton, 1997; Sage, 1995). With the increasing complexity of systems, purpose-build computer design software or design "tools" have emerged to assist engineers during design specification.

Bahill and Dean (2007), citing Bahill and Gissin (1998), mention that systems engineering is a process-driven discipline for addressing a customer's needs. The production of a "problem statement" is critical for meeting these needs. Bahill and Dean describe how a problem statement is addressed by requirement definitions to ensure that a system is designed to meet the customer's needs. The process defined by Bahill and Dean incorporates system modelling for refining stated requirements (Bahill & Dean, 2007, pp. 20).

Some form of model is produced by engineers during design specification. Krick (1969, p. 67) mentions that system engineers can employ models to facilitate communication through meaningful visual feedback, to forecast events and perform simulations, or assist in training procedures.

Bar-Yam (2003) provides a historic view of failed systems engineering projects and the probable causes of their failures. In the case of systems engineering for computer systems, the analysis by Le Lann (1996) is of relevance. Le Lann concludes that the disaster that was the Ariane 5 launcher, which devastated France's space programme, was a direct result of a systems engineering failure concerning hardware specifications.

1.1.2 Model-based Systems Engineering

The OMG (2006) mentions that systems engineers typically employ a multitude of tools and techniques to assist in their projects. The Model-based Systems Engineering (MBSE) concept aims to satisfy the simulation, prediction and analysis requirements of systems engineering projects through the development of a system model (Wymore, 1993). Stated customer requirements are translated and a model is produced to enable system configurations and "performance parameters" to be generated (Jansma & Jones, 2006).

Estefan (2007) conducted a survey of the foremost MBSE methodologies and found that the following methodologies incorporate the SysML for systems modelling: Object-oriented Systems Engineering Method (OOSEM), Telelogic Harmony-SE and IBM Rational Unified Process for Systems Engineering (RUP SE) for Model-Driven Systems Development (MDSD).

1.2 *Research Purpose and Significance*

According to Kobryn (2004), multiple UML 2.0 "language dialects" were developed for modelling software as a direct consequence of vendors choosing to implement only certain areas of the UML specification. Kobryn and Mueller et al. (2006) consider this to be a hindrance to the acceptance of the UML for modelling and this problem may also apply to the SysML (Kobryn, 2004, p. 7).

Compared to the UML, the SysML is a recent introduction to modelling languages. Because of its immaturity, prospective system modellers may be inclined to query its support by various modelling tool vendors. In this case, an appraisal of a candidate's support for the language would be a useful investigation. The first phase will be conducting such an appraisal. The compliance guide from the specification document that is used by this research provides an adequate benchmark for this phase. Namely, the OMG SysML 1.0a Final Adopted Specification (FAS) is this document.

Holt and Perry (2006, p. 4) point out that even prior to the specification and ratification of the SysML standard, software vendors were declaring that their modelling products were SysML compliant. This raises the issue of modelling products providing only partial compliance to the SysML standard.

A useful investigation would involve determining if each candidate tool presents a different representation of the model of a physical system. Also, this investigation can benefit from the results and materials of the first phase. The second phase is concerned with such an investigation.

This research may assist studies concerned with the compatibility of SysML models that are interchanged between different SysML modelling software. Reliable SysML model

interchange between language implementations is a concern of the "SysML PlugFest" project (Denno, 2006).

1.3 Research Questions

The aim of this research is to answer the following questions:

1. To what extent do current SysML modelling tools implement key parts of the OMG SysML 1.0 language specification?
2. Can a model of a physical, real-world system be represented consistently between the current SysML modelling tools?

1.4 Glossary of Terms

ADTF	Analysis and Design Task Force
CASE	Computer-aided Software Engineering
class	"A description of a set of objects that share the same attributes, operations, methods, relationships, and semantics" (OMG, 2001, p. 5).
DAS	Draft Adopted Specification
FAS	Final Adopted Specification
FTF	Finalisation Task Force
HSUV	Hybrid-powered Sports Utility Vehicle (OMG, 2006)
INCOSE	International Council on Systems Engineering

MBSE	Model-Based Systems Engineering
MDA	Model-Driven Architecture
MDSD	Model-Driven Software Development
MDSDWG	Model-Driven System Design Working Group
Metaclass	"A class whose instances are classes. Metaclasses are typically used to construct metamodels" (OMG, 2001, p. 11).
metamodel	"A model that defines the language for expressing a model" (OMG, 2001, p. 11).
MOF	Meta Object Facility
OCL	Object Constraint Language
OO	Object-Oriented
OOSEM	Object-Oriented Systems Engineering Method
OMG	Object Management Group
profile	A package containing a customisation of model elements for a specific domain. Constraints, stereotypes and tagged value definitions are used as "extension mechanisms" (OMG, 2001, p. 15).
RFP	Request for Proposal

RUP SE	Rational Unified Process for Systems Engineering
SoC	System-on-Chip
SMT	SysML Merge Team
SST	SysML Submission Team
SysML	Systems Modelling Language
UML	Unified Modelling Language
XMI	XML Metadata Interchange
XML	Extensible Markup Language

2 Literature Review

In order to elaborate subject matter related to this research, this section will discuss the concept of "systems modelling" and two significant modelling languages.

2.1 *Systems Modelling*

In the systems engineering domain, "system modelling" concerns the capturing of information in the design phases of systems development (Muth, 2001). Systems engineers typically employ multiple methodologies, according to the OMG (2006, p. 1).

The increase in complexity of engineering solutions has influenced the use of "system modelling" as a way of managing this complexity (Muth, 2001, p. 2). Traditionally, systems engineering has been mainly a process-driven discipline, with the concepts of "system modelling" and "object-orientation" from the software industry breaking new ground.

Modelling languages have emerged as a means for practitioners from diverse disciplines to model engineering problems. Languages such as the UML and the SysML were developed for the software engineering and systems engineering disciplines, respectively.

2.2 *The Unified Modelling Language*

The UML is an object-oriented (OO) modelling language for specifying, visualising and constructing software systems (Booch, Rumbaugh, & Jacobson, 2005; Selic, 2006). Its name signifies that, as a language, it draws upon several existing modelling concepts. These consisted of refined modelling notations and OO methods.

UML was adopted by the OMG in 1996 and since then, it has undergone a major revision to version 2.0 (Selic, 2006). Today, it remains the leading open industry standard notation for software development. The current official version of the UML at the time of this writing is 2.1.1, which was released in 2007.

The UML has been instrumental in several studies (Mueller et al., 2006; Vanderperren & Dehaene, 2005a; Yves & Wim, 2006) exploring the effectiveness of OO concepts in systems engineering. These studies typically employ the UML profile mechanism.

2.2.1 Profiles: UML Extensibility

One way of extending the capabilities of the UML is to use its "profile mechanism". This method effectively creates a language "dialect". It "tailors" the UML "metamodel" to align it with the concepts and semantics of a particular domain (OMG, 2007e, p. 13). To elaborate on what a "metamodel" is, Kühne (2006) provides a more formal definition. As an analogy, Kühne, citing Seidewitz (2003), states that in linguistics a "metamodel is a specification model for which the systems under study being specified are models in a certain modeling language". Metamodelling is to models what "metamathematics" is to mathematics: an application of the methods of a subject to the subject itself.

The UML's capabilities have been extended to specialised domains and application areas, such as system-level electronics design for System-on-Chip (SoC) projects (Mueller et al., 2006, p. 75).

"Tagged values", "constraints" and "stereotypes", are elementary in creating profile-based UML extensions (Mueller et al., 2006). These "meta" elements exist within the UML "superstructure", a structure consisting of modelling constructs for the user (OMG, 2007d, p. 1).

Mueller et al. states that "stereotypes are specific metaclasses (classes in the metamodel), tagged values are standard attributes of metaclasses...Constraints are semantic conditions or restrictions and can be applied to stereotypes." (2006, p. 74). Most constraints are expressed using OMG's Object Constraint Language (OCL).

Standards forms of UML stereotypes exist and are titled using guillemets ("«" and "»" chevrons affixes). For example, the "«import»" stereotype is used to indicate an importation relationship between "package" elements. "«entity»" is used to indicate a business object and "«derive»" is used to indicate a derivation between model elements (OMG, 2007d).

"Class" and "Port" are standard forms of UML metaclasses. The SysML, a UML language extension, employs stereotypes such as "«block»" and "«flowPort»" for specifying, respectively, Block and FlowPort elements.

2.3 *The Systems Modelling Language*

The SysML is a domain-specific language for systems engineering and it draws upon OO concepts from software engineering (OMG, 2006; Wang, Birla, & Neema, 2006). It is currently gathering considerable interest for systems modelling (Ganesan & Prevostini, 2006; Goering, 2006; Hause, Thom, & Moore, 2005; Kobryn, 2004; McGinnis, Huang, & Wu, 2006; Sibbald, 2006; Y Vanderperren & Dehaene, 2005a; Viehl, Schönwald, Bringmann, & Rosenstiel, 2006; Yves & Wim, 2006).

In a discussion about systems engineering for product lifecycle management, Bock (2005, p. 124) states that "systems engineering is currently hampered by a lack of a standard language for coordination across the product lifecycle and across disciplines involved in product development". SysML is intended to address the need of the systems engineering community for a standardised design language that incorporates the

capabilities of the UML (OMG, 2003, p. 22). An example of a SysML diagram is shown in figure 2.1.

The UML is capable of being extended using its profile mechanism, integrated with other OMG technologies and incorporated into a Model-Driven Architecture or MDA (Balmelli, Brown, Cantor, & Mott, 2006; OMG, 2003). According to the OMG (2003, p. 22), adopting the OMG's MDA initiative will require the establishment of capable system modelling frameworks, tools and methods for incorporating models created using legacy technologies or from specialised systems engineering domains.

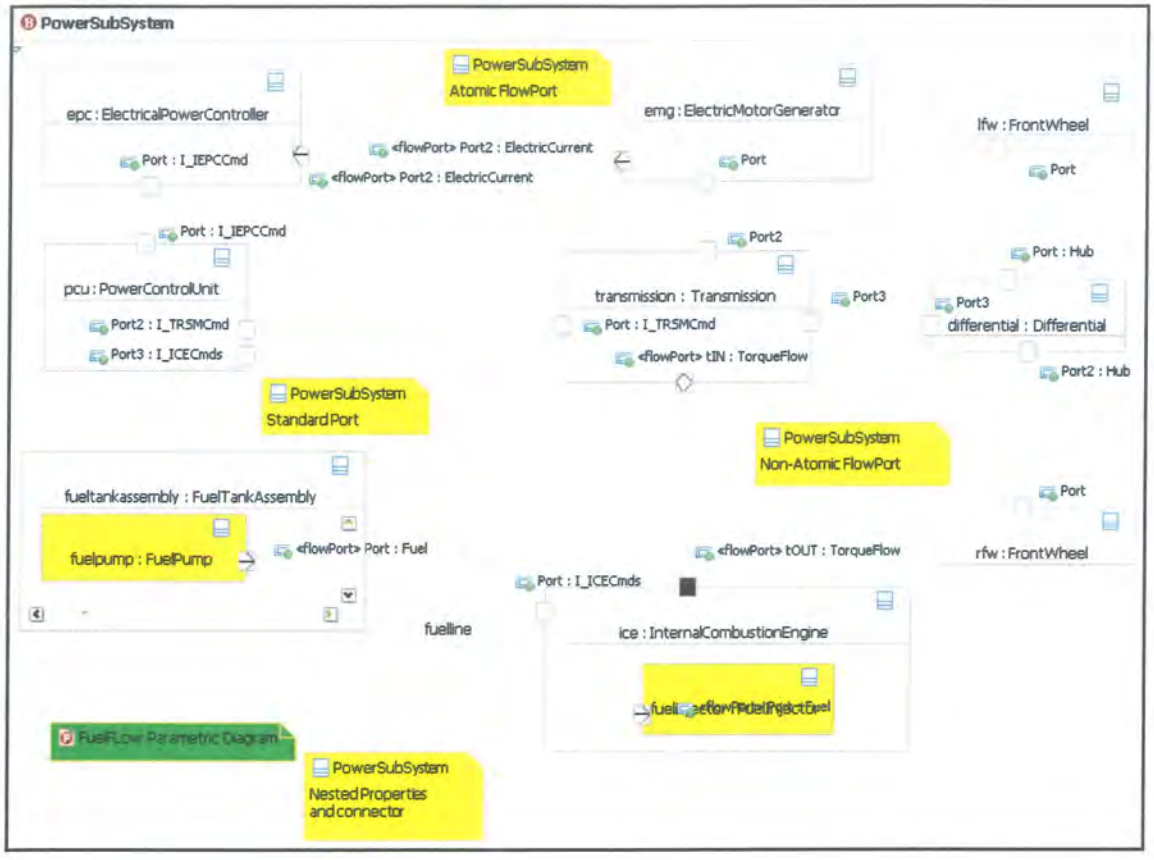


Figure 2.1: An example of a SysML Diagram using the EmbeddedPlus SysML Toolkit.

Figure 2.1 is an example of a SysML "internal block" diagram as shown in the FAS. It shows the layout of the power subsystem within an automobile. This diagram type builds upon its ancestor, the UML "composite structure" diagram, and focuses on the internal structure of a "block", which is a definition of a system feature (OMG, 2006). It

represents structure using "properties" (large rectangles), "ports" (small squares attached to Blocks) and "connectors" (lines connecting two symbols together).

SysML provides the additional ability to specify "flow ports" on blocks and properties, which are points of item transfer. "Atomic" flow ports can carry one item flow and "non-atomic" flow ports can carry several item flows. Atomic flow ports are shown in figure 2.1 and an arrowhead is used to indicate their direction. Figure 2.1 also contains "conjugated" flow ports that are shown as black squares, which are flow ports with "flow properties" with reversed directions.

The SysML was created especially to address a Request for Proposal (RFP) created through collaboration between the OMG and the Model Driven System Design Working Group (MDSDWG) of INCOSE. A summary of the SysML language specification's development is depicted in figure 2.2.

After this research began, the available specification of SysML 1.0 was released in September of 2007. The evaluation framework of this research is based on the OMG SysML version 1.0a FAS, which was released in July of 2006.

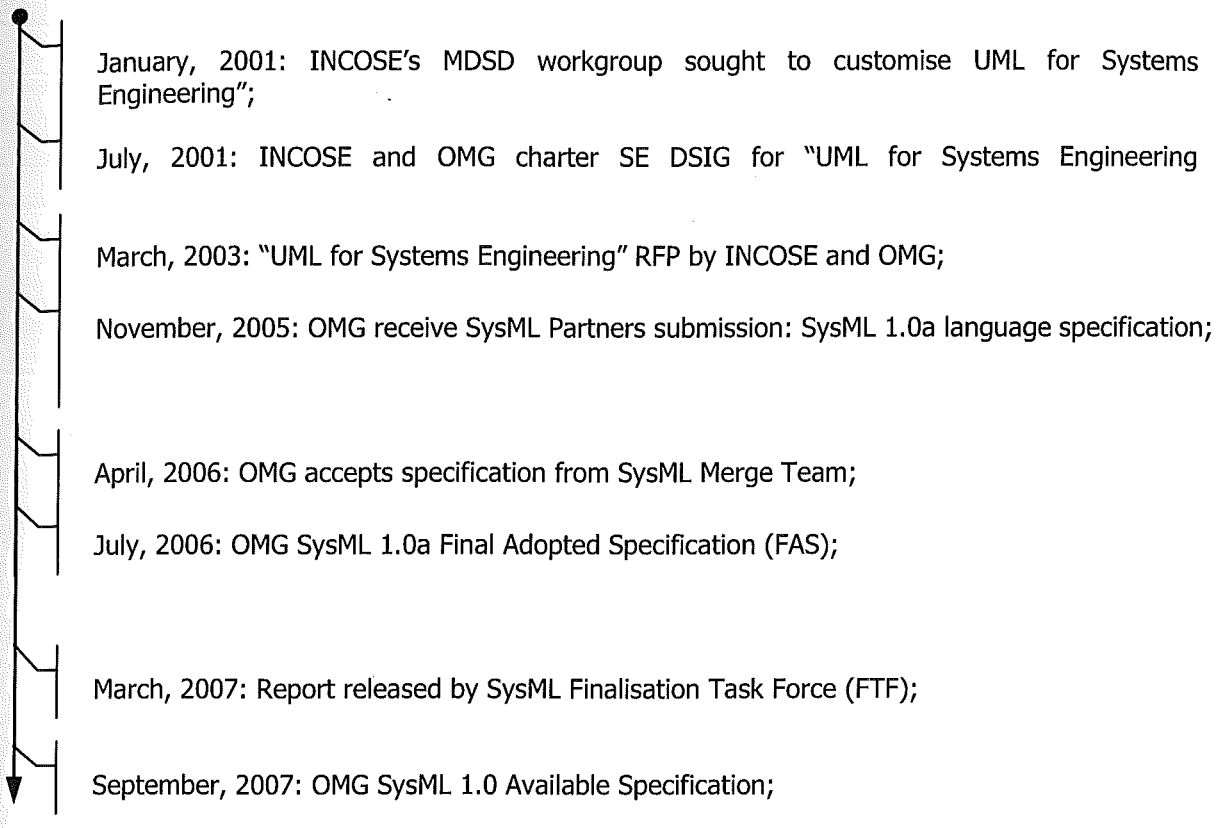


Figure 2.2: Timeline of the SysML's development (OMG, 2007c).

The SysML is an extension of the UML (see figure 2.3). Both of these modelling languages are inherently reliant on the concept of a "metamodel" (Kühne, 2006). In this context, a metamodel provides the user with a collection of tools, predefined rules and constraints for creating models based on a particular modelling language (Pidcock, 2003, pp. 9). Pidcock states that a metamodel can be defined as a model for a particular domain of interest. The design of this research will focus on the definition of the SysML 1.0a metamodel from the FAS.

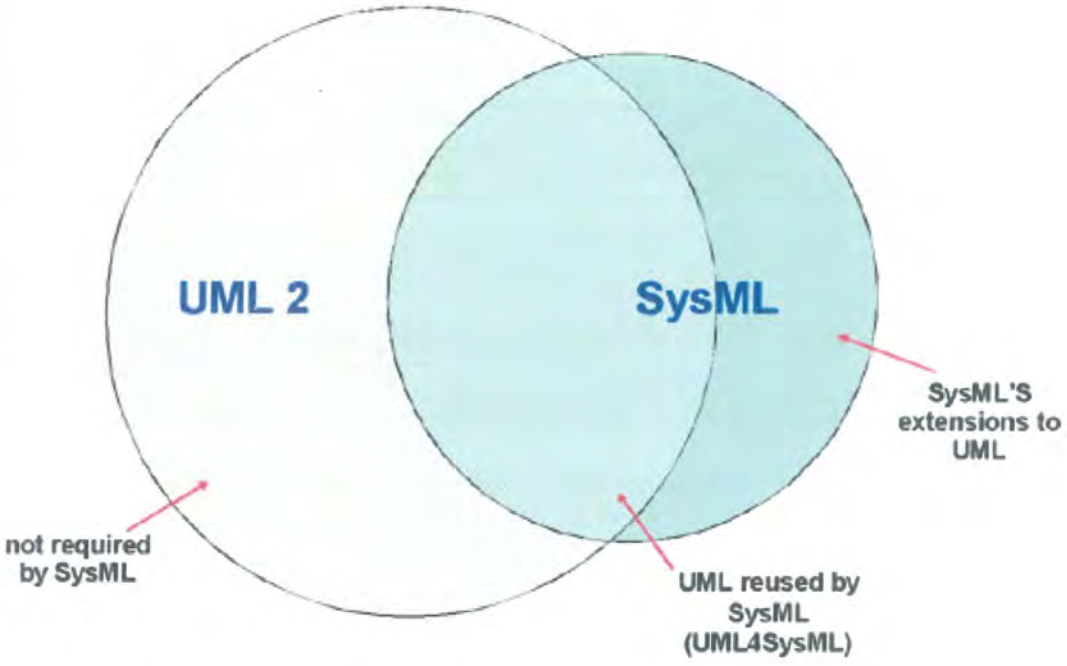


Figure 2.3: SysML extending the UML 2 (OMG, 2007c).

Given that there are several other alternative software systems, techniques and languages for modelling, including the UML: Why would someone choose to use the SysML? There are several reasons:

- It provides a systems engineering perspective by means of centralising information from multiple disciplines into a single system representation (Hause et al., 2005);
- It incorporates open standards, such as standards for diagram and model (or "metadata") interchange (2005);
- It is intended to be vendor neutral and compatible with modelling tools that support its notation (Wang et al., 2006, p. 55);
- It facilitates implementation for product vendors who are already acquainted with the current UML standard as it extends and reuses many of the UML's systems engineering concepts (OMG, 2006; Willard, 2007); and

- Its specification was developed with input from several tool vendors, organisations, systems engineering firms and United States (U.S) government divisions, all of which form the "SysML Development Team" (OMG, 2006, pp. 4-5).

Several studies exist (Colombo, Bianco, Lavazza, & Coen-Porisini, 2006; Ganesan & Prevostini, 2006; Jansma & Jones, 2006; McGinnis et al., 2006; Viehl et al., 2006) that either present approaches for applying the SysML or demonstrate its design capabilities for engineering projects.

The next section will discuss methods for conducting software evaluations of CASE tools.

2.4 CASE Tool Evaluation

Several studies (Kitchenham, Linkman, & Law, 1997; Kornecki & Zalewski, 2003; LeBlanc & Korn, 1994) report on software evaluations for improving software development projects within organisations. In each of these studies, an evaluation framework was devised.

Several different approaches exist for researchers and practitioners who are designing and conducting software evaluations, including CASE tool evaluations, for various purposes. For example, Kornecki and Zalewski (2003) conduct a quality assessment of design tools for developing on-board airborne software systems. These systems are relied upon for their real-time, safety-critical capabilities. An airborne software guideline named "RTCA DO-178B" is used to direct the evaluation, which is a quality-based assessment of programming and design tools.

LeBlanc and Korn (1994) perform a more general evaluation to that of the more recent works of Kornecki and Zalewski (2003) or Juric and Kuljis (1999).

The principles in Vessey, Jarvenpaa and Tractinsky (1992), cited by Juric and Kuljis (1999), appear consistent with several related studies. Both works elaborate on the "methodology companion", the concept of a CASE tool aiding the developer's adoption of a methodology. According to McClure, cited in Vessey, Jarvenpaa and Tractinsky (1992, p. 1), these companions enhance the utility of development tools. These studies employ a similar, tabulated attribute framework for comparing modelling tools.

Vessey, Jarvenpaa and Tractinsky (1992) focused on assessing the ability of tools to satisfy the operation of a particular software development methodology. Several tools were assessed on their successful execution of processes and delivery of products.

3 Theoretical Framework

This section will begin by briefly describing issues related to this research. It will state research questions and provide an overview of the research design and describe methods for addressing those questions. Finally, alternative research approaches will be discussed along with a quantification technique.

Since the SysML is rapidly being considered as a suitable modelling language for systems engineering projects (McGinnis et al., 2006), prospective and participating practitioners may be concerned about its support by available modelling software. In relation to the evolving language standard, the maturity of the available modelling tools and how each vendor has implemented the language are important issues. Furthermore, a greater issue is how a non-compliant implementation may affect the products being developed.

There appears to be no literature available that investigates the language compliance of modelling tools for the SysML. According to the websites of the "SysML Forum" (2007) and a number of software vendors (""EmbeddedPlus SysML Toolkit for the IBM RSDP"," 2007; ""Magicdraw SysML Plugin"," 2007; ""Sparx Systems - MDG Technology for SysML"," 2007), there exists several tools claiming support for the SysML 1.0.

The "compliance" statement in the FAS states that compliant software must have addressed all "applicable compliance points as stated in the specification" and that the OMG is the authority for granting compliance certification (OMG, 2006). These are relevant issues because the objective of the SysML is to extend the concepts from the UML into the domain of systems engineering.

3.1 *Research Design*

This research was designed to answer the following research questions:

1. To what extent do current SysML modelling tools implement key parts of the OMG SysML 1.0 language specification?
2. Can a model of a physical, real-world system be represented consistently between the current SysML modelling tools?

The questions will be answered through an evaluation of candidate tools consisting of two phases: one being a comparative evaluation and the other, a qualitative evaluation based on a real-world model. These phases are depicted in figure 3.1.

Both phases are explained in detail in the subsections ahead, together with definitions of the research methods employed and considerations of alternative methods. Galliers (1990) discusses various interpretive and empirical methods that are appropriate for this field of research.

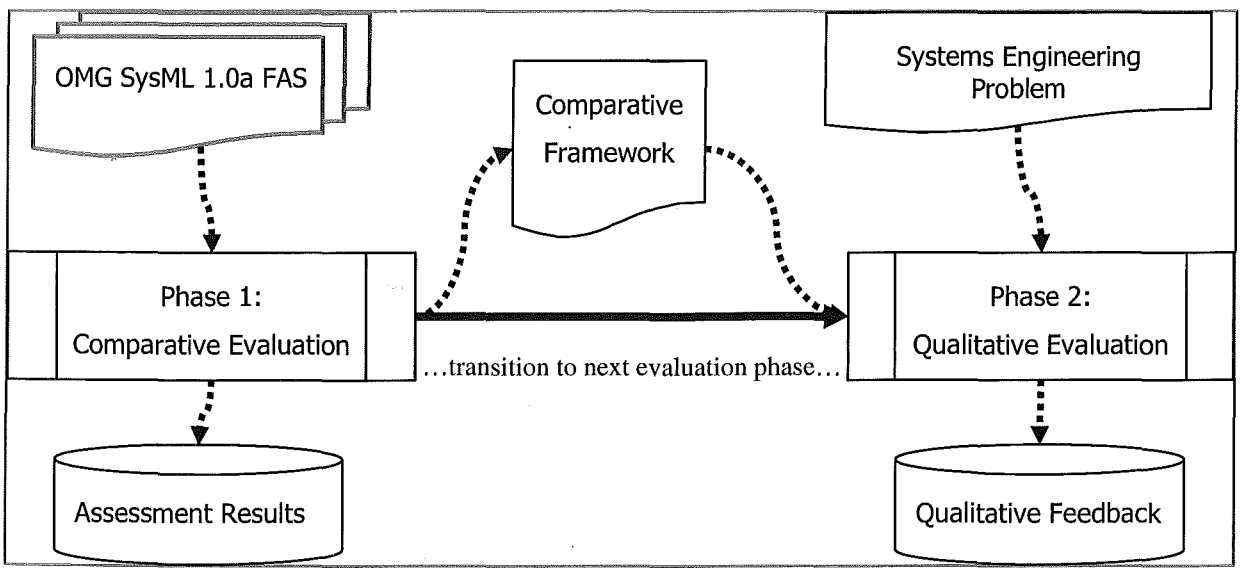


Figure 3.1: The processes and dependencies of the research design.

3.1.1 Comparative Evaluation

The comparative evaluation aims to answer the first research question by identifying the compliance of candidate modelling tools to the SysML standard using the OMG SysML 1.0a FAS as a template. It was inspired by the research design of Juric & Kuljis (1999) and Law's (1988) advice on comparative methods.

Juric and Kuljis (1999) performed an evaluation of CASE tools based on their implementation of "rules" derived from the UML language. Their study focused on creating an evaluation instrument based on the "rules" of the UML in order to assess the language constraints of UML modelling software. Each rule was earlier derived from the UML 1.1 standard by Juric (1998). The study evaluated two CASE tools and examined their validation or "checking" mechanisms to determine to what degree they supported the UML (Juric & Kuljis, 1999). These rules were considered as a set of diagramming constraints, enforceable by "methodology companions", such as CASE tools.

For the conduct of comparative research techniques in software development scenarios, Law (1988) provides context, procedures, guidelines and suggestions and also elaborates on several important contributing factors. When assessing methodologies, an analytical framework of some description must be considered (Law, 1988, p. 19). Such a framework consists of "features" or "attributes". Their consistency and level of detail can present potential management problems for the evaluator, such as complexity and misinterpretation (Law, 1988, p. 31).

Lundell and Lings (2002), citing Kitchenham and Jones, state that a comparative evaluation of CASE tools requires an evaluation framework. Also, they mention that the

3.1.1 Comparative Evaluation

The comparative evaluation aims to answer the first research question by identifying the compliance of candidate modelling tools to the SysML standard using the OMG SysML 1.0a FAS as a template. It was inspired by the research design of Juric & Kuljis (1999) and Law's (1988) advice on comparative methods.

Juric and Kuljis (1999) performed an evaluation of CASE tools based on their implementation of "rules" derived from the UML language. Their study focused on creating an evaluation instrument based on the "rules" of the UML in order to assess the language constraints of UML modelling software. Each rule was earlier derived from the UML 1.1 standard by Juric (1998). The study evaluated two CASE tools and examined their validation or "checking" mechanisms to determine to what degree they supported the UML (Juric & Kuljis, 1999). These rules were considered as a set of diagramming constraints, enforceable by "methodology companions", such as CASE tools.

For the conduct of comparative research techniques in software development scenarios, Law (1988) provides context, procedures, guidelines and suggestions and also elaborates on several important contributing factors. When assessing methodologies, an analytical framework of some description must be considered (Law, 1988, p. 19). Such a framework consists of "features" or "attributes". Their consistency and level of detail can present potential management problems for the evaluator, such as complexity and misinterpretation (Law, 1988, p. 31).

Lundell and Lings (2002), citing Kitchenham and Jones, state that a comparative evaluation of CASE tools requires an evaluation framework. Also, they mention that the

composition of the framework may depend on the motivation and knowledge of those contributing to the evaluation.

In preparation for an evaluation within an organisation, Law (1988) follows a three stage process for removing unsuitable comparative elements. This process requires screening attributes and candidates on two levels of detail in order to check for qualities needed by the comparison.

Jansma and Jones (2006, p. 5) describe the "SEA project", which "evaluated a number of systems engineering tools against a specified set of criteria and attempted to evaluate each tool using a real-world scenario". McGinnis, Huang and Wu (2006, p. 1882) employ a "small scale example" model for their modelling and simulation experiment.

To assist in assessing the compliance of candidate tools to specific areas of the OMG SysML 1.0a FAS, a framework was put together for this evaluation. It is composed of rules derived from "abstract constraints" defined within various subsections of the FAS (OMG, 2006).

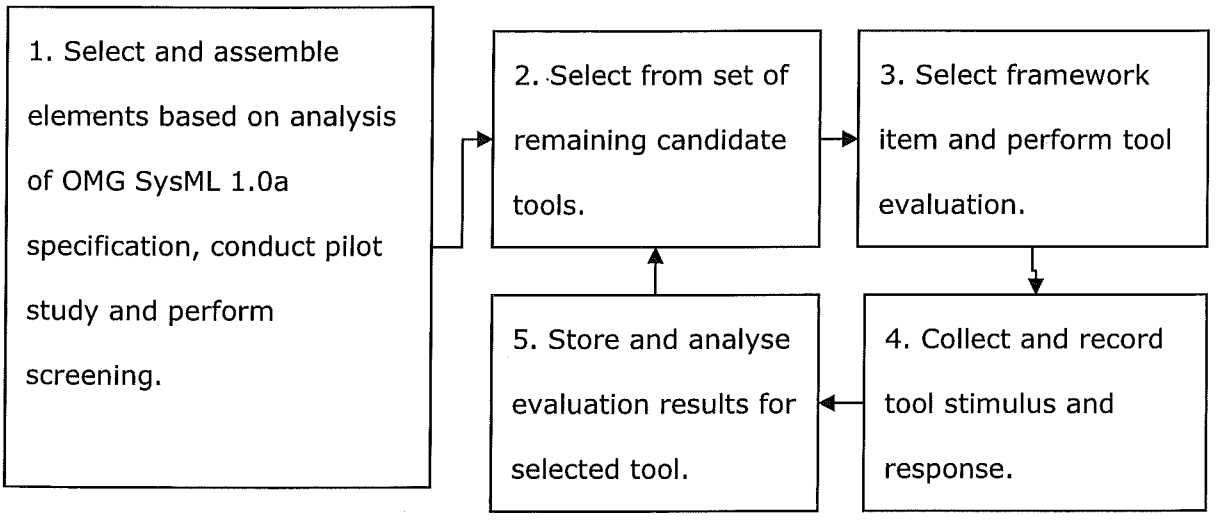


Figure 3.2: Comparative evaluation process for the candidate tools.

3.1.2 Quantification

A weighted ranking technique will be employed in the comparative evaluation. Law (1988) mentions that weighted ranking techniques can be used to analyse the results of an evaluation.. McDermid (1990, p. 346) discusses techniques for quantifying the outcome of an evaluation and conducting estimations. One technique is called "weighted ranking with levels" and McDermid describes how a classification system can be introduced to aid a comparison of candidates. In his discussion, McDermid describes attribute counting, ranking based on scores, weighted ranking and weight ranking based on levels, all of which are particularly applicable to the results analysis of this research. Law (1988, p. 106) stresses that numerical scores obtained from assessed characteristics have limited significance and are merely indicators formed using a predefined scale.

3.1.3 Interpretive Evaluation

The second question relates to SysML's ability to model a physical system and if this ability is realised any differently in each candidate. This phase addresses that question and assesses the ability of the research candidates to consistently represent a physical, real-world system. A qualitative evaluation of each candidate will require the modelling of such a system. Several studies on CASE tool evaluation have used this approach as part of their research design.

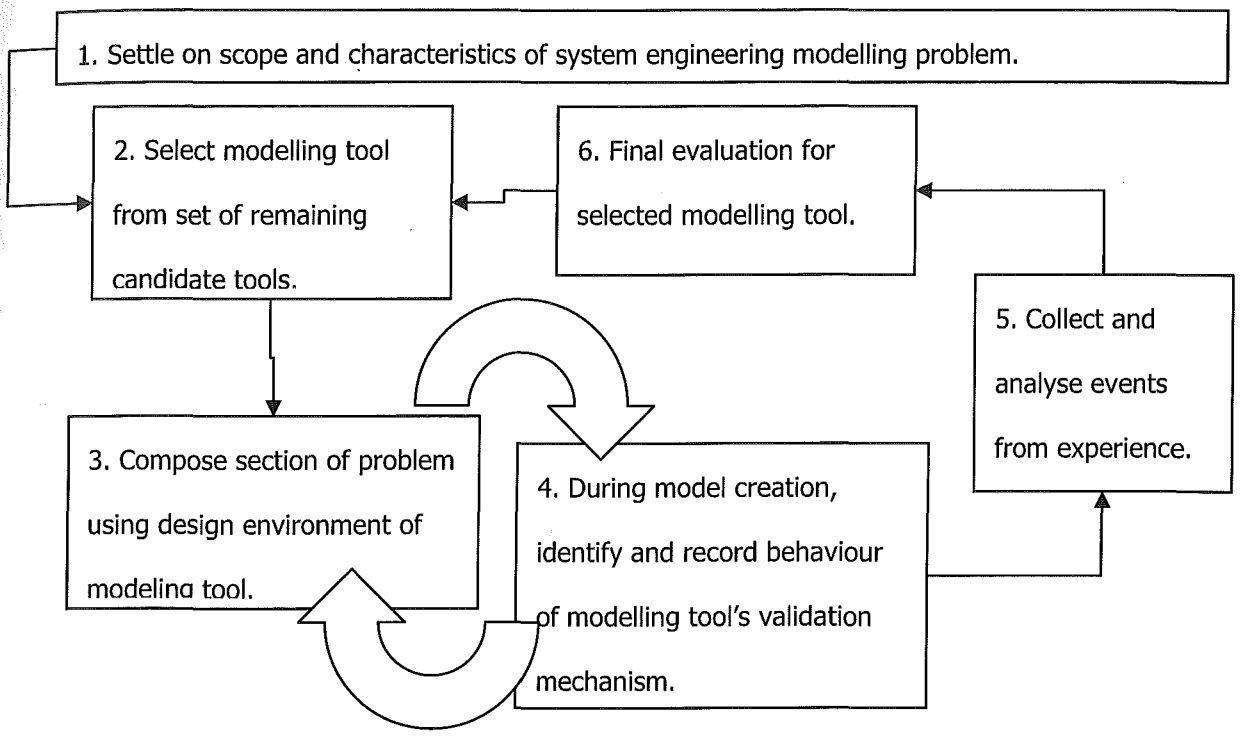


Figure 3.3: Interpretive research process using an ideal systems engineering problem.

This evaluation conducted exercises on each candidate by applying a stimulus and gathering the resulting, observable behaviour. After these behavioural data were recorded and compiled, they were analysed further.

This form of evaluation is significant for several reasons. Firstly, it offers a practical assessment of the SysML 1.0a support of several candidates using an exemplary systems engineering problem. Secondly, it applies the SysML to a system modelling problem that incorporates elements from a real-world systems engineering project. Thirdly, it incorporates an evaluation framework formed during the previous comparative evaluation, which serves as criteria for verifying the SysML 1.0a compliance of each candidate. Finally, among the available SysML literature, this is a unique study as it focuses only on the specific SysML extensions to the UML as documented in the FAS.

Galliers (1990) classifies descriptive and interpretive research as more modern approaches than empirical methods based on observation. They are described as appropriate ways to study methodologies or technologies (Galliers, 1990, p. 168). According to Galliers, a review of past studies can be carried out as part of interpretive research to enable theories and knowledge to be developed about a subject.

The approach of using an ideal problem as part of a candidate evaluation was inspired by the research of Floyd (1986). In Floyd's case, the problem was intended to be representative of a typical development problem for evaluating development methodologies for Information Systems (IS). Floyd admits that, despite its theoretical basis, the research problem benefited the researcher's knowledge and overall assessment. However, it must be noted that Floyd's research compared system development methodologies and not modelling tools.

3.2 *Alternative Approach: Experiments*

The research approaches mentioned were chosen for their suitability in answering the research questions. This section describes their suitability and reasons for excluding other methods for IS research (Galliers, 1990).

The research questions imply investigations concerning the identification of FAS compliance. They complement one another since they both question the compliance and modelling capabilities of each candidate.

The research method taxonomy of Galliers (1990) is shown in figure 3.4.

Approach	Key Features	Strengths	Weaknesses
Laboratory Experiments	Identification of precise relationships between chosen variables via a designed laboratory situation, using quantitative analytical techniques, with a view to making generalisable statements applicable to real-life situations.	The solution and control of a small number of variables which may then be studied intensively.	The limited extent to which identified relationships exist in the real world due to oversimplification of the experimental situation and the isolation of such situations from most of the variables that are found in the real world.
Field Experiments	Extension of laboratory experiments into the real-life situations of organisations and/or society.	Greater realism; less artificial/sanitised than the laboratory situation.	Finding organisations prepared to be experimented on
Surveys	Obtaining snap shots of practice, situations or views at a particular point in time (via questionnaires or interviews) from which inferences are made (using quantitative analytical techniques) regarding the relationships that exist in the past, present and future.	Greater number of variables may be studied than in the case of experimental approaches. Description of real world situations. More easy/appropriate generalisations.	Likely that little insight obtained re. the causes, processes behind the phenomena being studied. Possible bias in respondents (cf. self-selecting nature of questionnaire respondents); the researcher, and the moment in time which the research is undertaken.
Case Studies	An attempt at describing the relationships which exist in reality, usually within a single organisation or organisational grouping.	Capturing 'reality' in greater detail and analysing more variables than is possible using any of the above approaches.	Restriction to a single event/organisation. Difficulty in generalising, given problems of acquiring similar data a statistically meaningful number of cases. Lack of control of variables. Different interpretations of events by individual researchers and stakeholders.
Forecasting, Future Research	Use of such techniques as regression analysis and time series analysis, or the delphi method and change analysis, to extrapolate/deduce likely/future possible events or impacts.	Provision of insights into likely future occurrences in situations where existing relationships may not hold true in the future. Attempts to deal with the rapid changes taking place in IT and their impacts on individuals, organisations and society in general.	Complexity and changing relationship of variables under study. Lack of real knowledge of future events. Scenarios are not 'true' pictures of the future but enable decisions re. reactions in different 'futures'. Dependence on precision/relevance of past data and expertise of scenario builders. Possibility of self-fulfilling prophecies.
Simulation, Game/Role Playing	An attempt at copying the behaviour of a system which would otherwise be difficult/impossible to solve analytically by the generation/introduction of random variables.	Provision of an opportunity to study situations that might otherwise be impossible to analyse.	Similar to experimental research in regard to the difficulties associated with devising a simulation that accurately reflects the real world situations.
Subjective, Argumentative (CF. Phenomenology, Hermeneutics)	Creative research based more on opinion/speculation than observation, thereby placing greater emphasis on the role/perspective of the researcher. Can be applied to existing body of knowledge (reviews) as well as actual/past events/situations.	Useful in building theory that can subsequently be tested. Creation of new ideas and insights. Recognition that the researcher will interpret what is being studied in a particular way. Contributes to cumulative knowledge.	Unstructured, subjective nature of research process. Despite making the prejudice of the researcher known, there is still the likelihood of biased interpretations, a problem which is confounded by the time at which the research is undertaken.
Action Research	Applied research where there is an attempt to obtain results of practical value to groups with whom the research is allied, while at the same time adding to theoretical knowledge.	Practical as well as theoretical outcomes most often aimed at emancipatory outcomes. Biases of researcher made known.	Similar to case study research, but additionally places a considerable responsibility on the researcher which objectives are at odds with other groupings. The ethics of the particular research are a key issue.

Figure 3.4: A taxonomy for IS research methods from Galliers (1990, p. 166).

The first research question implies a measurement of compliance. It does not focus on why candidates have attained their level of FAS compliance or what processes were involved in their development. More accurately, it concerns an investigation into what compliance variations may exist between candidates. Therefore, scientific approaches based on relationship identification, such as experiments, would not have sufficed (Galliers, 1990, p. 161).

The second research phase is driven by a leading question of the fundamental modelling capability of each candidate. This phase benefits from the results and element framework produced by the first phase. These products can assist in identifying and confirming the modelling capability of each candidate more precisely. Also, the framework elements involved in performing systems modelling with each candidate can be identified. The second research question does not demand a design that involves the use of experimental methods, interactions with groups, event prediction, system simulation or a form of creative research.

Experimental approaches are intended for establishing relationships between controlled and independent variables. Since they are intended to explain phenomena, they are not appropriate approaches for this form of research. The simplified nature of experiments presents a complication that cannot accommodate for the amount and complexity of the observable and controlled variables required to confirm a candidate's FAS compliance.

Perhaps an experiment would be best applied to explaining the behaviour of each candidate tool during modelling exercises.

Experiments and field experiments (their "extension" to real world situations) both rely on the establishment of a controlled environment for variable isolation. If an experimental method was considered, it would be difficult to identify independent and

dependent variables for each experiment. Also, the language implementation of each candidate is not representative of a single phenomenon and was probably developed under unique circumstances. Therefore, their application would have been ineffective for determining FAS compliance (Galliers, 1990).

Even though both research questions rely on gathering empirical evidence, a candidate assessment is a more suitable approach.

3.3 *Alternative Approach: Case Study*

Benbasat, Goldstein and Mead (1987, p. 370) outline the characteristics of case research and its suitability to IS research. It is effective in capturing practitioners' knowledge and studying their use of industry practices and processes within their environment. Also, a "case approach is an appropriate way to research in an area in which few previous studies have been carried out" (Benbasat et al., 1987, p. 370), which, according to Benbasat et al., is appropriate for the constantly evolving field of IS.

Although using a case study's results to support a generalisation is difficult (Galliers, 1990, p. 162), they could be incorporated into the evidence gathering phase of a cross-case study. Yin (1981) describes the case-comparison and case-survey approaches of cross-case studies for deriving limited generalisations across cases.

However, the case study approach was not considered for this research since it is suited to capturing occurrences within a suitable organisation (Galliers, 1990). The modelling problem for this research was developed in an academic setting and a qualitative evaluation approach was chosen to address it.

4 Resources

The resources used to conduct this research will be described in this section.

For meeting its objectives, this research requires certain hardware and software resources. These included a general-purpose computer and a set of candidate modelling tools. In regards to any significant intellectual materials, the first phase of this research relied on the SysML 1.0a FAS for forming an element framework. The second phase also relied on the FAS as a reference for an ideal model.

It was essential that the computer system used in this research was capable of operating each candidate modelling tool. Also, a network connection to the Internet was required for obtaining each candidate software product for the evaluation.

The use of each modelling tool required the provisioning of time-limited licenses from the following software vendors: NoMagic Incorporated, Sparx Systems Pty Ltd and EmbeddedPlus Pty Ltd. The evaluation required these licenses to provide the full-functionality of each candidate tool.

5 Limitations

This section will elaborate on the various constraints applied to this research and the reasons behind their application. These constraints consider the scope of the SysML's language syntax, SysML software implementations and the model required for the second evaluation phase.

5.1 Language Syntax

To keep within the time constraints of this research, limited language compliance was considered for the evaluations. The framework was populated with elements using only the abstract constraints from selected FAS sections. Only the sections stipulating SysML extensions to the UML were considered. Furthermore, a criterion for abstract syntax compliance involving the interchange of models using the XML Metadata Interchange (XMI) standard was not considered due to time constraints.

To further limit this research, three tools will be considered in the evaluation phases. Also, the scope of modelling tool types will be broad. A "tool" will be considered as any software application capable of modelling diagrams using the SysML notation.

5.2 Tool Implementations

This research design considers the discussions by Kobryn (2004) and Mueller et al. (2006) on the nature of UML implementations by vendors. The researcher acknowledges that the implementations under evaluation may vary in language compliance to the SysML and that the outcomes of the evaluations may not necessarily indicate a lack of support for that language.

5.3 *Model for Qualitative Evaluation*

Due to the unavailability of a real-world problem for this project, a sample problem provided with the SysML FAS was chosen as an alternative. It is an example of a specification that is under development for a Hybrid-powered Sports Utility Vehicle (HSUV). It demonstrates SysML's fundamental modelling capabilities (OMG, 2006, p. 171) using appropriate diagrams for specifying requirements, structure, behaviour and operational constraints. It is also used to elaborate on how the SysML addresses the "UML for Systems Engineering" RFP (OMG, 2003, p. 44). The figures contained within the FAS for the sample problem are suitable for the purposes of this research's qualitative evaluation.

To ensure consistency between the two phases of this research, only the parts of the sample problem that utilised SysML extensions to the UML were considered in this evaluation.

6 Comparative Evaluation

This section will elaborate on the comparative evaluation phase of this research by firstly describing the research candidates, the assessment framework and methods for collecting data. Next, a pilot study will be described along with the methods used for screening the framework's elements. This will be followed by a discussion of the results from the evaluation and its analysis phases.

The comparative evaluation relied on testing the descriptions contained within each element of the framework. A nominal scale consisting of "true", "false", and "partial" values was used to measure each evaluated element (Sarle, 1997). A true value for an element signifies that the evaluation found the candidate to fully satisfy the requirements stipulated in the element's description. A partial value for an element indicates that the candidate failed to satisfy at least one of the element's requirements. If none of an element's requirements were satisfied during the evaluation, the candidate is afforded a false result for that element.

In a discussion about the ISO standard for evaluating CASE tools, Lundell (2002) states that an organisation will go through four evaluation phases: "preparation; evaluation and selection; pilot project; and transition" (2002, p. 382).

6.1 Evaluation Candidates

The research design relies on an appropriate set of modelling tools for gathering empirical evidence. At a minimum, the vendor for each modelling tool must have declared at least some support for the SysML modelling language. Three applications that met this requirement were chosen as candidates for the evaluation (see table 6.1).

Table 6.1:

Candidate modelling tools with their SysML extensions.

Application Name	Extension Name
Sparx Systems Enterprise Architect 7.0	MDG SysML Technology Add-In 6.5
IBM Rational Software Architect 7.0.0.3	EmbeddedPlus SysML Toolkit 2.0.0.2
Magicdraw UML Enterprise Edition 14.0 EAP (Beta 1)	Magicdraw SysML Plugin 1.1

For brevity, this research will refer to "Sparx Systems Enterprise Architect 7.0" as the "Sparx Systems tool", "IBM Rational Software Architect 7.0.0.3" as the "EmbeddedPlus tool" and "Magicdraw UML Enterprise Edition 14.0 EAP (Beta 1)" as the "Magicdraw tool".

Each candidate consists of a modelling environment and an associated extension developed specifically for providing SysML language support. The modelling environment serves as a platform for the extension, which augments the environment by enabling SysML-specific functionality. This functionality includes the ability to create diagrams based on the eight SysML diagram types (OMG, 2006, p. 11) and to compose models using the SysML notation.

Certain facilities provided by the platform are crucial to the SysML-specific extension. For instance, in order to support the SysML notation, the environment should support the UML 2.1 metamodel, which is usually in the form of a language profile.

The next section elaborates on the element framework used in the comparative evaluation.

6.2 *Comparative Framework*

During the comparative evaluation phase, an element framework was formed to assist in assessing each candidate. It was composed using content from the FAS considered

suitable for creating a set of language rules. Namely, the content consisted of extensions to the UML that were specific to the SysML. These extensions consisted of constraints for modelling elements that govern their usage. Also, the framework incorporated any attributes that were specified for an element. An attribute consists of a property or feature of a stereotype in the SysML (OMG, 2006, p. 11).

Each framework element is associated with a description about the FAS constraint or attribute that they are based on. These descriptions were extracted from sections of the FAS that detailed SysML extensions to the UML. They described either an "attribute" or a "constraint" associated with a UML element for the SysML. Ambiguous or repeated descriptions were rejected during the gathering process to maintain the element framework's accuracy and consistency.

The decision to incorporate a framework into this research was based on the suggestions from Law (1988) and the OMG (2006). Law emphasises a process based on the definition of quality attributes for the purpose of assessing a methodology, method or tool's suitability to organisational requirements (Law, 1988, pp. 38-39). Law mentions complexity issues arising when managing a framework composed of different attribute levels. Also, Law states that a direct comparison of candidates on an attribute level is more conclusive that the use of a scaling system.

In a document submitted to INCOSE that reviewed a proposal for the SysML from the SysML Submission Team (SST), "compliance levels" are described as being of great importance to users of the SysML (Skipper, Estefan, & Shames, 2006). The review mentions that interoperability between SysML tools may be compromised by varying compliance levels. The SST made an earlier, noteworthy comment on the architecture of version 0.9 of the SysML, saying that "ambiguity affects vendor ability to implement" (SST, 2005).

For those developing SysML implementations, the FAS provides compliance guidelines and these were used to form the evaluation design. Hause, Thom, & Moore (2004) claim that these guidelines were adopted from the UML 2.0 specification as a means for assessing such implementations.

An ideal framework, the OMG SysML 1.0a FAS itself, is incorporated into this comparative evaluation. The evaluation framework was constructed using the attributes and constraints of each element within the FAS document. This form of evaluation is comparable to a "macro-evaluation", which is an evaluation of a tool's quality by focusing on the use of a tool for design work, as described by Kornecki & Zalewski (2003).

The FAS describes each element and, where applicable, it elaborates on their required attributes and associated language constraints. The "constraints" part of each description stipulates mandatory behaviours for a particular element. Each description elaborates on the appearance of elements shown on a diagram. How each element is rendered on a diagram can be significantly influenced by the modelling tool and the software preferences set by the end user.

The FAS measures compliance in terms of "abstract" and "concrete" syntax using the package hierarchy of the language (OMG, 2006). Syntax definitions are shown in table 6.2. Abstract syntax concerns the parts of the SysML specification that describes the language's meta-model, its rules and constraints. The interchange of models based on the XMI standard is also part of abstract syntax compliance. Concrete syntax consists of the language's graphical notation. SysML compliance is twofold, requiring software to be compliant with its own metamodel and that of its underlying UML dependent. In the case of the SysML 1.0a FAS, it is a UML 2.1 dependent (OMG, 2006).

Table 6.2:

Syntax definitions for SysML FAS compliance (OMG, 2006, p. 15).

Compliance	
Abstract Syntax Compliance	**Concrete Syntax Compliance**
Metaclasses, stereotypes, model libraries,	Notation for Diagram Elements
Constraints and Structural Relationships	Notation for Diagram extensions
Model exchange using XMI schema	Supported Diagram types

The FAS defines the "meaning of compliance" for modelling tool vendors seeking to implement the notation and semantics of the SysML 1.0a. It states that implementations are required to comply with both the concrete notation and the abstract syntax of the SysML and the fundamental UML4SysML metamodel. The UML4SysML metamodel consists of the UML elements that are reused by the SysML. This relationship is shown in figure 2.3.

Since the SysML is dependent on an underlying UML 2 implementation, compliance with its metamodel also requires compliance with the UML4SysML metamodel. This metamodel is depicted in figure 6.1 using a structure of packages separated into three levels.

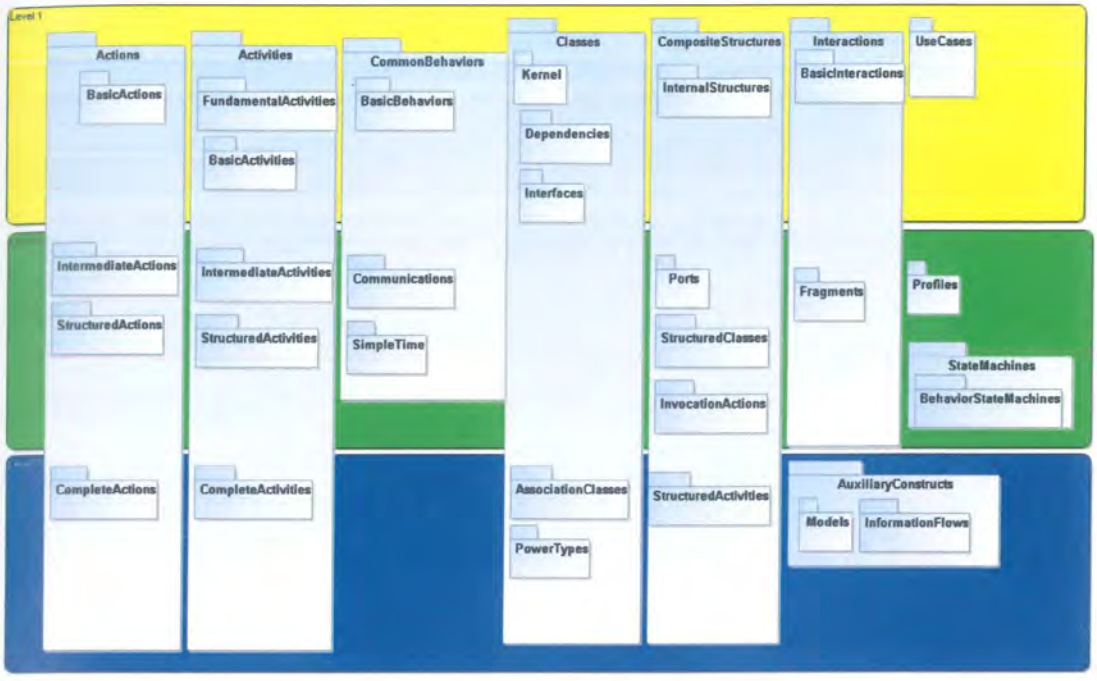


Figure 6.1: Packages of the UML4SysML metamodel that SysML depends on (OMG, 2006, pp. 13-14).

OMG (2006) states the UML4SysML subset metamodel is divided into three UML compliance levels and the SysML reuses the packages from these levels. These UML compliance levels are composed of the metamodel's metaclasses, which are organised into a package structure.

According to OMG (2006), level one of the UML4SysML metamodel contains UML essentials for modelling Actions, Activities, Use cases and Interations. Level two contains facilities for creating language profiles and modelling state machines. Level three provides the SysML metamodel with package and information flow concepts.

Levels are used to indicate interdependencies between packages, which are manifestations of interdependencies between metaclasses at different levels. Metaclasses within higher level packages are extensions of metaclasses from lower level packages and are therefore dependent on them. OMG (2006) states that compliance with the metaclasses of a SysML package, with the exception of certain

elements, requires compliance with a certain level of the UML4SysML compliance structure. Certain elements, such as the "Probability" stereotype element may be dependent on packages at multiple compliance levels and those dependencies are passed on to the packages containing those elements, such as the "Activities" package (OMG, 2006).

OMG (2006) provides a guide for structuring an evaluation for a language implementation and uses a nominal scale to measure the compliance of each element. This scale consists of the discrete values "YES", "NO" and "partial" as defined in OMG (2006).

For an evaluation result, a "YES" value is used to signify that all the requirements contained within an element's description have been satisfied. A "NO" value indicates that no stated requirements have been addressed for a particular element. For indicating that a quantity of requirements less than the element's total was satisfied, a "PARTIAL" value may be used, together with an elaboration in the form of field notes and comments. The aforementioned definitions are used by this research's evaluation. However, the names "true" and "false" are used instead of "YES" and "NO", respectively.

The FAS contains examples of "compliance statements" and "feature support statements" for demonstrating a qualitative evaluation based on the language's "compliance levels". Most of the sections within the FAS contain a "UML Extensions" subsection, which describes the SysML extensions to the underlying UML metamodel in terms of concrete and abstract constraints (OMG, 2006).

Section five of the FAS provides compliance guidelines for language implementers (OMG, 2006). These guidelines contain examples for structuring the results of

evaluations that test for complete, partial or non-existent compliance of the SysML notation and meta-model.

The approach of employing a framework as set of language rules for evaluating candidate was based on two studies from the field of CASE tool evaluation (Juric & Kuljis, 1999; Vessey et al., 1992).

The evaluation phase of this research considers the definition of abstract syntax compliance within the FAS. It concentrates on the constraints, attributes, stereotypes, meta-classes, model libraries and abstract relationships stipulated in each area of the specification. However, the exchange of model information based on the XMI schema will not be considered during the evaluation.

Apart from being useful references on the appearance and usage of SysML diagram notation, neither the "Concrete Syntax Examples" nor the "Usage Examples" were considered adequate sources of language rules from the specification. Instead, from each chapter, the "Description", "Constraints" and "Attributes" parts for each SysML extension were considered, since they provided sufficient detail of the language's semantics, its boundaries and how its elements may be configured. Through this, a set of language rules could be formulated.

These rules were later used to exercise the validation function of each candidate tool using deliberately constructed models. These exercises were designed to trigger a tool's validation function and gather a response as a way of interrogating the tool's language implementation. Model validation is available in most current CASE tools for software modelling.

After performing enough exercises on a tool to address each rule, an assessment was made of its language implementation. The following section will describe the methods used to collect data during these exercises.

6.3 *Data Collection*

During a candidate evaluation, several forms of data may be obtained from a tool and examined in order to afford a result for an evaluated framework element. These data include the state of the modelling tool's graphical user interface (GUI), exported XMI data, notifications received as part of user feedback from an active validation mechanism and the tool's documentation.

According to the OMG (2006, p. 217) the XMI 2.1 standard allows software to exchange model information for any language defined using the Meta Object Facility (MOF). Since the UML is based on MOF, tools may serialise and exchange SysML models using XMI.

Certain elements required the examination of XMI information exported by a candidate to adequately determine the existence of language elements and to supplement the evaluation results.

Table 6.3:

Data Collection methods for modelling tool candidates.

Evidence	Advantages	Disadvantages
GUI	Can provide direct user feedback. Can provide indicators in the way of restrictions, notifications, in various ways on the required use of modelling functions.	Vendor dictates what information is provided by the GUI façade.
Validation System	Can provide feedback on a model's integrity and validation status.	Covers modelling constraints.
XMI	Exposes the attributes, constituents and relationships for all elements contained within a model.	Difficult to interpret given the serialised form of the model.
Software Documentation	Can contain comments from developers and guides regarding validation rules for SysML models.	Is limited to information about the tool's implementation. May not be consistent with the tool or complete.

A significant issue for the comparative evaluation phase was finding an adequate method of gathering evidence during each candidate assessment. In most cases, the candidate's GUI obscures the view of the metamodel implementation or language profile. The XMI exportation is an alternative data collection method and many tools provide a facility for doing so. XMI exports may be closely inspected and compared to the framework's requirements.

```
<?xml version="1.0" encoding="UTF-8"?>
<uml:Model xmi:version="2.1"
xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
xmlns:uml="http://schema.omg.org/spec/UML/2.1.1"
xsi:schemaLocation="http://schema.omg.org/spec/UML/2.1.1
http://www.eclipse.org/uml2/2.0.0/UML"
xmi:id="_jHPRJ4VfEdyr4ciwwtrtNA" name="SysML Model">
  <packageImport xmi:type="uml:PackageImport"
xmi:id="_jHPRKIVfEdyr4ciwwtrtNA">
    <importedPackage xmi:type="uml:Model"
href="http://schema.omg.org/spec/UML/2.1.1/uml.xml#_0"/>
  </packageImport>
  <packageImport xmi:type="uml:PackageImport"
xmi:id="_jHPRKYVfEdyr4ciwwtrtNA">
    <importedPackage xmi:type="uml:Model"
href="pathmap://SYSML_MODELLIBS/Blocks.uml#_1L5dsL93EdqaocM3Gp-xwg"/>
  </packageImport>
  <packageImport xmi:type="uml:PackageImport"
xmi:id="_jHPRKoVfEdyr4ciwwtrtNA">
    <importedPackage xmi:type="uml:Model"
href="pathmap://SYSML_MODELLIBS/Activities.uml#_1L5dsL93EdqaocM3Gp-
xwg"/>
  </packageImport>
  <packagedElement xmi:type="uml:Class"
xmi:id="_jHPRK4VfEdyr4ciwwtrtNA" name="Subject"/>
  <profileApplication xmi:type="uml:ProfileApplication"
xmi:id="_jHPRLIVfEdyr4ciwwtrtNA">
    <xmi:Extension extender="http://www.eclipse.org/emf/2002/Ecore">
      <eAnnotations xmi:type="ecore:EAnnotation"
xmi:id="_jHPRLYVfEdyr4ciwwtrtNA"
source="http://www.eclipse.org/uml2/2.0.0/UML">
        <references xmi:type="ecore:EPackage"
href="http://schema.omg.org/spec/UML/2.1.1/StandardProfileL2.xmi#_yzU5
8YinEdqtvbnfB2L_5w"/>
      </eAnnotations>
    </xmi:Extension>
    <appliedProfile xmi:type="uml:Profile"
href="http://schema.omg.org/spec/UML/2.1.1/StandardProfileL2.xmi#_0"/>
  </profileApplication>
</uml:Model>
```

Figure 6.2: An XMI source file

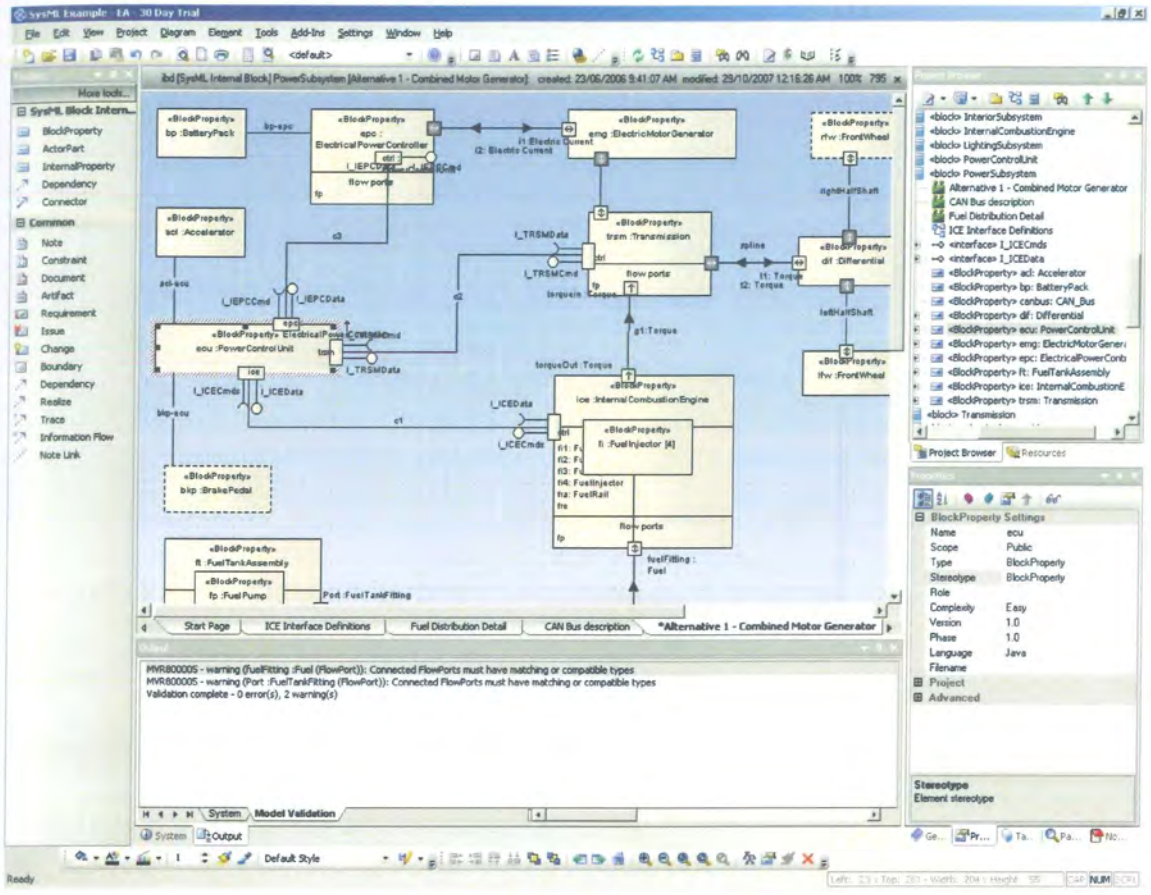


Figure 6.3: A screenshot of a typical GUI for a UML modelling application.

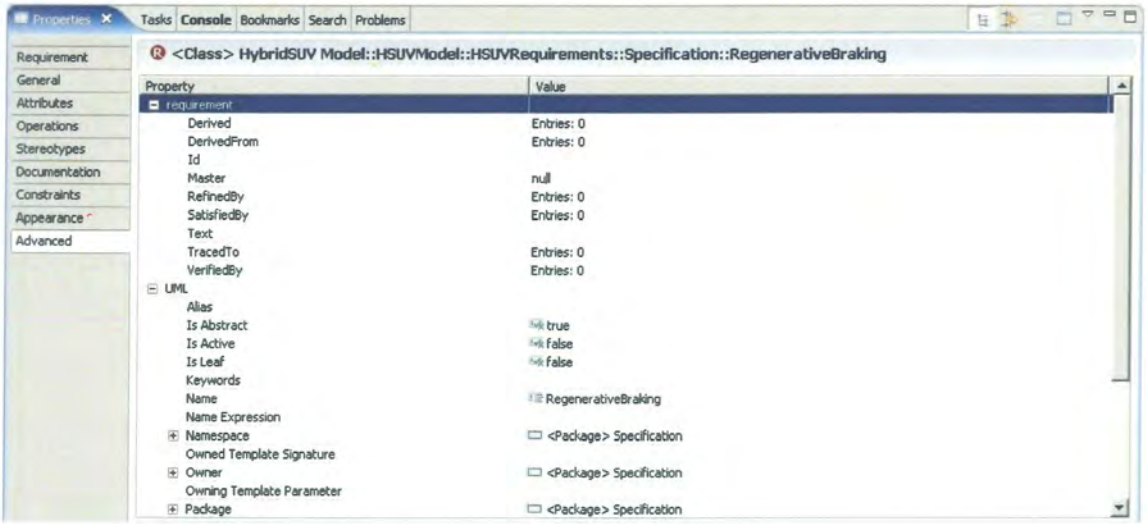


Figure 6.4: A property listing for a currently selected element

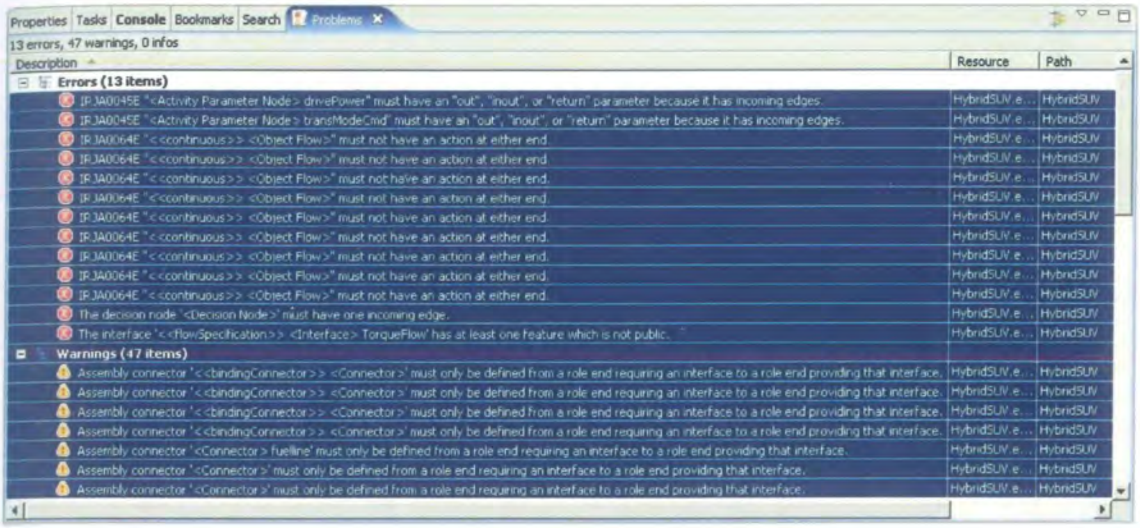


Figure 6.5: Validation output from IBM Rational Software Architect 7.0

A simple example of an XMI file is shown in figure 6.2. Figure 6.4 and figure 6.5 are examples of various GUI features used for data collection. These examples were taken from IBM Rational Software Architect 7.0.

Figure 6.3 shows the standard GUI of Sparx Systems Enterprise Architect 7.0. It displays properties for the currently selected element, the current diagram, the active model and the output of the model validation feature.

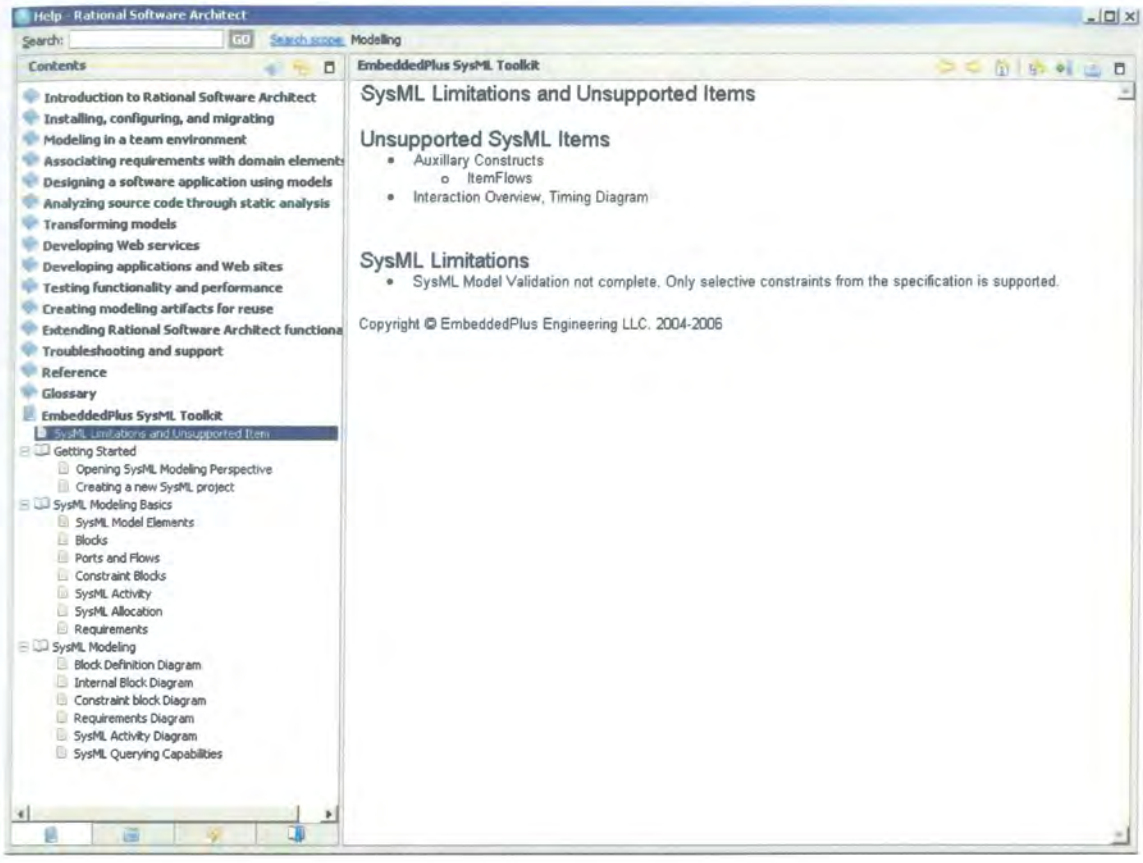


Figure 6.6: Documentation for the EmbeddedPlus SysML Toolkit.

The next section discusses a pilot study for testing the initial comparative framework.

6.4 The Pilot Study

A pilot study was conducted with the Sparx Systems tool as part of the evaluation process. The initial trial (pilot study) allowed the effectiveness of the element framework to be tested and for evidence to be gathered for subsequent attribute screening.

The following section elaborates on the screening procedures applied as a result of the pilot study.

6.4.1 Framework Screening

The evaluation conducted during the pilot study produced an element framework that was too finely-grained (greater than 110 elements). A screening rationale was devised to ensure that any redundant, optional or ambiguous framework elements were deleted in order to obtain the minimum number of elements that can discriminate. It is available in appendix A (see section 11).

After a second attempt at applying the framework to a candidate tool, further screening was performed. In order to avoid applying such a detailed framework to each candidate tool, elements from each section of the framework were examined and deleted using a three phase screening process.

6.4.1.1 Screening Stage One: Constraints and Attributes

This stage of the screening process filtered any elements that were not part of the rationale for the evaluation framework. Those elements that were not created using abstract constraints from selected areas of the FAS were eliminated.

6.4.1.2 Screening Stage Two: Optional, Repeating and Ambiguous Elements

This stage of the screening process filtered any elements that were optional, ambiguous or found to be repeatedly testing the same language concept. A number of constraints from the specification relied on "semantic variation" and these were removed in order to improve the accuracy of subsequent evaluations. Elements created from substitutable abstract constraints were also removed.

6.4.1.3 Screening Stage Three: Unsupported UML 2.x features

To ensure that each tool could be tested evenly for equivalent SysML constraints, screening was applied to compensate for unsupported UML 2.1 features that were

common to each candidate. Elements were discarded if they were dependent on an underlying UML 2.1 profile element that was not available in all candidate modelling tools.

The "ParameterSet" element for Activity diagrams is an example of such a dependent element. No candidate modelling tool involved in the evaluation process provided a way of applying parameter sets to an activity element. Constraints that depended on the existence of this element were not included in the final evaluation framework. The initial (pilot) framework is available in appendix B (see section 12).

The results and analysis phases of the evaluation will be described in following sections.

6.5 Results

The evaluation demonstrated that the candidate tools were compliant with at least fifty percent of the framework's elements. The results obtained showed that a greater majority of attributes were satisfied than the majority of rules. Between the candidates, ten "partial" element results were obtained.

In terms of satisfying framework elements, the Magicdraw tool covered the most number of elements and the least was covered by the Sparx Systems tool.

The evaluation proved that the use of multiple data collection methods was necessary. This is due to the fact that each candidate tool offers a unique user interface that offers different insights into the modelling environment. For example, the Sparx Systems and EmbeddedPlus tools present the user with different levels of detail for a SysML model under development.

To elaborate on the data obtained during the evaluation, a few artefacts will be presented from each candidate.

Table 6.4:

Sample of results for the Sparx Systems tool

Category	Description	Result	Notes	Feedback
Rule	The ownedAttribute property must not have a value defined.	TRUE		
Rule	«requirement» stereotyped classes are unable to have association relationships.	TRUE	Validation Rule: MVR800013	MVR800013 - error (ARandomSystemRequirement (Requirement)): A requirement cannot participate in associations
Rule	«requirement» stereotyped classes are unable to have generalisation relationships.	TRUE	Validation Rule: MVR800013	MVR800013 - error (ARandomSystemRequirement (Requirement)): A requirement cannot participate in generalizations

Figure 6.7 and Figure 6.8 are samples of artefacts that were obtained from the EmbeddedPlus tool. The connector labelled "B1toC1" in figure 6.7 corresponds to the "ownedConnector" element in figure 6.8.

Table 6.4 is a sample of results for the Sparx Systems tool. It shows a description and result for each element together with the evaluator's field notes and any feedback obtained from the tool's model validation feature.

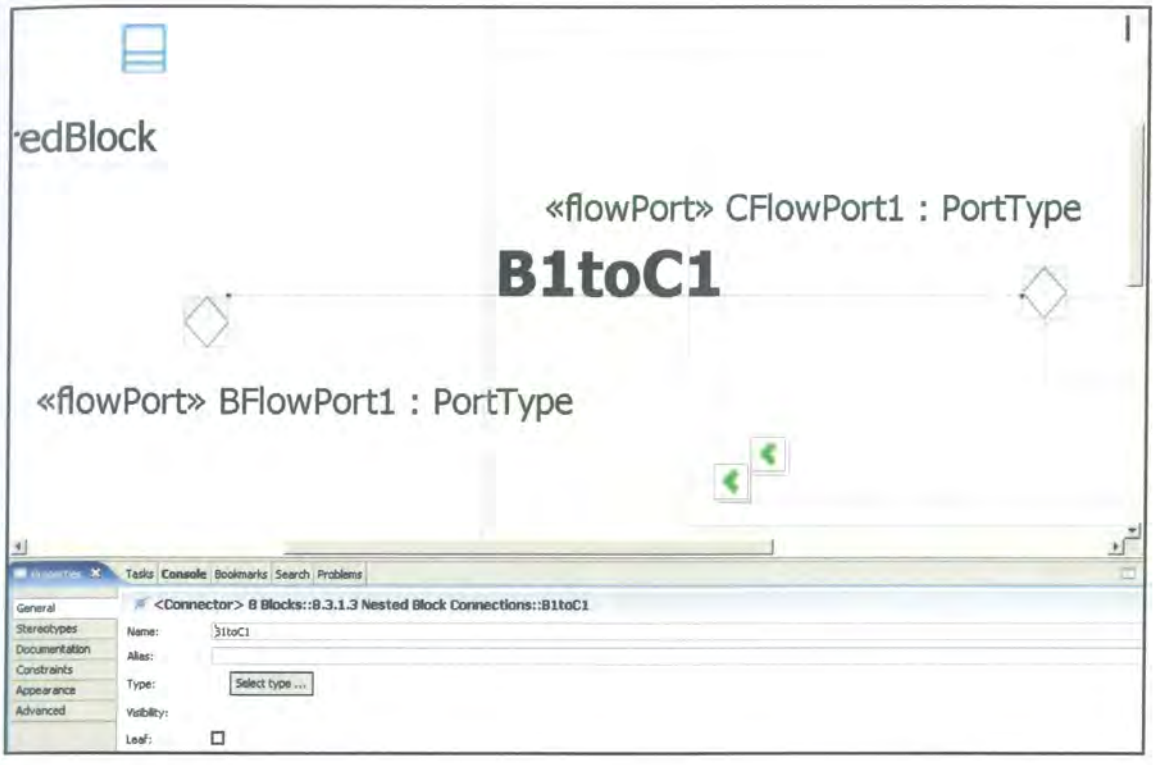


Figure 6.7: A screenshot artefact showing a connector within a diagram.

```
    <ownedConnector xmi:type="uml:Connector"
xmi:id="_jU1lBofXEdy4BarytrIuxQ" name="B1toC1" kind="assembly">
      <end xmi:type="uml:ConnectorEnd"
xmi:id="_jU1lB4fXEdy4BarytrIuxQ"
partWithPort="_jU1k7ofXEdy4BarytrIuxQ" role="_jU1k4ofXEdy4BarytrIuxQ">
        <xmi:Extension
extender="http://www.eclipse.org/emf/2002/Ecore">
          <eAnnotations xmi:type="ecore:EAnnotation"
xmi:id="_jU1lCIfXEdy4BarytrIuxQ" source="PROPERTYPATH">
            <details xmi:type="ecore:EStringToStringMapEntry"
xmi:id="_jU1lCYfXEdy4BarytrIuxQ" key="PROPERTYPATH"
value="_XUDJIGtDEdypqMJ2QXMmBw;_estYwGtDEdypqMJ2QXMmBw;"/>
          </eAnnotations>
        </xmi:Extension>
      </end>
      <end xmi:type="uml:ConnectorEnd"
xmi:id="_jU1lCofXEdy4BarytrIuxQ" role="_jU1lGYfXEdy4BarytrIuxQ">
        <xmi:Extension
extender="http://www.eclipse.org/emf/2002/Ecore">
          <eAnnotations xmi:type="ecore:EAnnotation"
xmi:id="_jU1lC4fXEdy4BarytrIuxQ" source="PROPERTYPATH">
            <details xmi:type="ecore:EStringToStringMapEntry"
xmi:id="_jU1lDIfXEdy4BarytrIuxQ" key="PROPERTYPATH"
value="_XUDJIGtDEdypqMJ2QXMmBw;_a5jawWtEEdypqMJ2QXMmBw;_dvMyMWtEEdypqM
J2QXMmBw;"/>
          </eAnnotations>
        </xmi:Extension>
      </end>
</ownedConnector>
```

Figure 6.8: An XMI source artefact corresponding to the connector in figure 6.7

Further interpretation of the raw results will be performed during the following analyses.

6.6 *Analysis*

This section will cover the analysis phases conducted on the comparative evaluation results. A modelling taxonomy for evaluation framework will be introduced in order to simplify the results. Also, a weighted ranking technique will be applied.

6.6.1 First Phase using Result Totals

In the first phase, totals were calculated for each tool based on the scale described in section 6.1. Table 6.5 displays the totals for the raw values gathered during each candidate's evaluation.

Law (1988, p. 44) mentions several methods for analysing results. According to Law, affording importance factors to attributes is subjective and may leave the analysis outcomes open to interpretation. Law recommends providing comments with every assessed framework attribute in order to provide qualitative feedback on each candidate evaluation. In this evaluation, comments were only provided with elements if they were not satisfied.

For each candidate, totals are calculated for each possible kind of result that is obtained during the assessment. To enhance the readability of the results, a percentage of language coverage for each candidate can be worked-out using the total number of elements that are satisfied.

Table 6.5:

Summary of evaluation results.

	Candidate		
Result	**EmbeddedPlus SysML Toolkit 2.0.0.2**	**MDG SysML Technology Add-In 6.5**	**Magicdraw SysML Plugin 1.1**
TRUE	71	64	88
%	62.3%	56.1%	77.2%
FALSE	39	48	22
%	34.2%	42.1%	19.3%
PARTIAL	4	2	4
%	3.5%	1.8%	3.5%
Note:	**Total number of elements is**		**114**

Assuming the total number of "true" values gained for each candidate is a direct measure of tool compliance, table 6.5 indicates that the MagicDraw tool ranks first, the EmbeddedPlus tool ranks second and the Sparx Systems tool ranks third.

However, this approach simply tallies results and does not consider the varying levels of importance that may be attributed to the framework elements. Also, the elements extracted from the various sections of the FAS are meant for different purposes. Subsequent approaches will explore the use of logical groupings and a weighed ranking technique as a way of addressing these shortcomings.

The next analysis phase considers dividing the evaluation results into exclusive groups and subtotalling their results as part of a more detailed analysis.

6.6.2 Second Phase using Groups

The second analysis phase involved aggregating the evaluation results using an appropriate taxonomy. Initially, this taxonomy consisted of the four SysML language "pillars", or aspects, described by OMG (2007c, pp. 10), namely: Structure, Behaviour,

Requirements and Parametrics. The use of these aspects enables a comparison to be made between candidates based on structural, behavioural, requirements and parametrics modelling constraints derived from the FAS. Figure 6.9 shows the "four pillars" of the SysML diagram taxonomy used to create the framework attribute categories (OMG, 2007c).

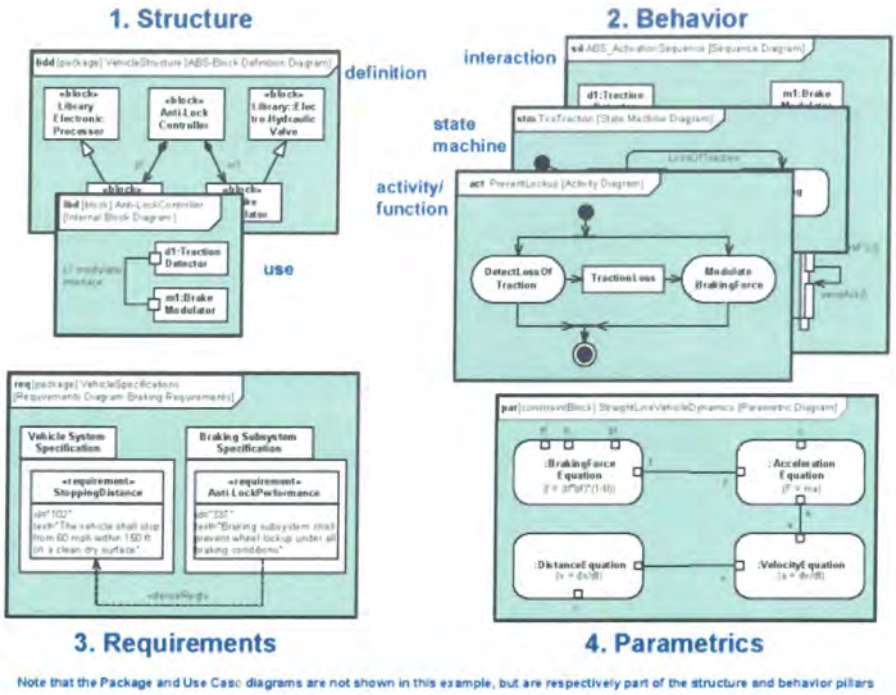


Figure 6.9: The modelling taxonomy of the SysML (2007c, pp. 11).

However, not all elements could be associated with a modelling aspect. Certain elements were derived from sections of the specification that defined intermediate modelling aspects, such as those for defining "allocation" relationships. Generic elements that are utilised by each of the aforementioned modelling aspects also exist. Therefore, two additional groups were needed to categorise these intermediate elements: "Model Element" and "Allocation".

This taxonomy was chosen over an alternative classification method based on the diagram type prescribed for each framework element. Impartial trials and examinations of the candidate tools identified a common inconsistency with each candidate. They

exhibited an inherent flexibility that does not force the user to use a specific branch of the modelling notation that is appropriate for a selected diagram type. For example, with the Magicdraw tool, one may compose a sequence diagram using notation for defining blocks and activities. Therefore, this inconsistency invalidates an element classification based on diagram type. Also, the classification would not have been appropriate for all elements, particularly the "cross-cutting" elements sourced from the "Allocations" section of the FAS. Instead, this analysis approach grouped the framework elements using a set of modelling aspects.

These groupings could facilitate a direct comparison between tools based on a particular aspect that is consistent with the FAS, such as the ability to model requirements. The differences between each result group may be observed in order to determine which groups contributed to the final score for each candidate. This approach considers each group to be of equal importance.

The framework elements were categorised based on the location in the FAS of their respective constraints and attributes. The FAS contains three main parts titled "Structural Constructs", "Behavioural Constructs" and "Crosscutting Constructs". The constraints and attributes from section seven formed the "Model Elements" group and sections eight and nine formed the "Structure" group. Sections ten, eleven, fifteen and sixteen formed the "Parametrics", "Behaviour", "Allocation" and "Requirements" groups, respectively.

As a measure of compliance, this analysis considers the total "true" values obtained for each group. It does not focus on the number of "false" values for each group since they merely represent the inverse of this approach. "Partial" values were not considered in this analysis due to their small number, which does not significantly contribute to the candidate rankings. The rankings for each group result are shown in table 6.6.

Table 6.6:

Group and overall rankings for each candidate.

	Candidate		
Group	EmbeddedPlus SysML Toolkit 2.0.0.2	MDG SysML Technology Add-In 6.5	Magicdraw SysML Plugin 1.1
Allocation	**2**	**2**	**1**
Behaviour	**2**	3	**1**
Model Element	3	**1**	**1**
Structure	3	**2**	**1**
Parametrics	3	**1**	**1**
Requirements	**1**	3	**2**
Rank	3	**2**	**1**

The ranks for each group in table 6.6 are based on the number of satisfied elements. A candidate's final rank was calculated as the mode of its group rankings. The grouped results are tabulated in table 6.7.

Table 6.7:

Summary of group results for the evaluations.

		Group						
Tool	**Result**	**Allocation**	**Behaviour**	**Model Element**	**Parametrics**	**Requirements**	**Structure**	**Grand Total**
Embedded Plus	TRUE	1	6	14	0	25	25	71
	FALSE	1	15	2	1	3	17	39
	PARTIAL	2	0	0	0	0	2	4
EmbeddedPlus Total		4	21	16	1	28	44	114
Magicdraw	TRUE	3	9	15	1	24	36	88
	FALSE	1	12	0	0	2	7	22
	PARTIAL	0	0	1	0	2	1	4
Magicdraw Total		4	21	16	1	28	44	114
Sparx Systems	TRUE	1	5	15	1	10	32	64
	FALSE	1	16	1	0	18	12	48
	PARTIAL	2	0	0	0	0	0	2
Sparx Systems Total		4	21	16	1	28	44	114

According to table 6.6, the EmbeddedPlus tool ranks third. However, table 6.7 indicates that, based on the total number of satisfied elements, the EmbeddedPlus tool places second. This difference is caused by the varying quantities of elements in each group.

This analysis phases considers each category to be equally important. The results show that the categories with the greater number of elements, "Behaviour", "Requirements"

and "Structure", influenced the final scores for each candidate the most.. "Parametrics" was the least influential category with only one element.

The candidate rankings have not altered since the previous analysis approach. However, the aspects that contribute to these rankings can now be observed more closely. Also, a selected group may be used to perform a direct comparison.

After an examination of the results from table 6.8, the following can be deduced:

- In regards to the "Allocation" group, the Magicdraw tool satisfied three times more elements than either the EmbeddedPlus or Sparx Systems tool;
- In comparison to the EmbeddedPlus tool, the Magicdraw tool satisfied more "Structure" group elements and less "Requirements" group elements;
- Given that the Sparx Systems tool placed before the EmbeddedPlus tool, it satisfied more elements from the "Model Elements", "Parametrics" and "Structure" section;
- The "Parametrics" group was satisfied by the Sparx Systems and Magicdraw tool. However, this group contains only one element and contributes little to a candidate's overall ranking;
- Out of the set of candidates, the Magicdraw tool satisfied the most elements from the "Allocation", "Behaviour" and "Structure" group;
- The lowest percentage of compliance originated from the "Behaviour" group; and
- The highest percentage of compliance originated from the "Model Element" group;

There are several possible factors that may contribute to the results in table 6.8 and the rankings in table 6.6. One major factor is the quantity of elements in each grouping.

Two critical categories were responsible for the ranking positions of the Sparx Systems tool and the EmbeddedPlus tool in table 5. These were the "Behaviour" and "Requirements" categories, with the EmbeddedPlus tool scoring highest overall in the latter category.

When comparing the results of the Sparx Systems tool and the Magicdraw tool, they are evenly matched on the "Parametrics" and "Model Elements" groups. The Magicdraw tool's results for the "Allocation", "Behaviour", "Structure", "Requirements" categories contributed significantly to its final mark. The Sparx Systems and Magicdraw tools addressed the "Parametrics" group, which required only a single element to be satisfied.

Another approach to the analysis is to afford a weighting factor to each group within the framework. The next section will investigate this approach.

6.6.3 Third Phase using Evenly Weighted Group Rankings

Law (1988) mentions how weighted ranking technique may be employed for producing an aggregated final result for an evaluation. McDermid (1990, p. 346) provides an example of this quantification method and mentions that results may be subdivided into classifications with associated weights as an aid to comparison. The weighted ranking technique relies on affording a weight for a set of elements based on a common property.

Weighting factors were applied to each group within the element framework. The reason behind this decision was due to the framework's constituents. Employment of the weighing technique as a means for interpreting the evaluation results raises a

significant question: how does one rate the importance of each framework element? By calculating the total number of elements satisfied by a candidate as its final score, one is assuming that all elements are equally important. The detail of the element framework, post-screening, presents a problem. Its immensity presents a complication when affording a weighting factor to each element within a group. Also, each group contains a different amount of elements ranging from 1 to 44 elements. McDermid (1990) mentions the importance of maintaining proportionality when applying weighting factors to elements.

In terms of this analysis, a weighting factor is a percentage indicating the significance of a group. A weighted result for a group is the product of the group's evaluation results and the weighting factor. A result for a particular evaluated group is calculated by multiplying the percentage of satisfied elements (those with a "true" value) with the group's weighting factor. Final scores were used in the weighted ranking example provided by McDermid (1990). In this analysis, a candidate's final score is calculated as a percentage by totalling the candidate's weighted results for each group.

The evaluation results using even weighting factors for each group are tabulated in table 6.8 and extrapolated in figure 6.10.

Table 6.8:

Summary of group results using even weighting factors.

Group	Weighting Factor	Candidate: EmbeddedPlus SysML Toolkit 2.0.0.2	Candidate: MDG SysML Technology Add-In 6.5	Candidate: Magicdraw SysML Plugin 1.1
Allocation	17%	4.2%	4.2%	12.5%
Behaviour	17%	4.8%	4.0%	7.1%
Model Element	17%	14.6%	15.6%	15.6%
Structure	17%	9.5%	12.1%	13.6%
Parametrics	17%	0.0%	16.7%	16.7%
Requirements	17%	14.9%	6.0%	14.3%
TOTAL		**47.9%**	**58.5%**	**79.9%**

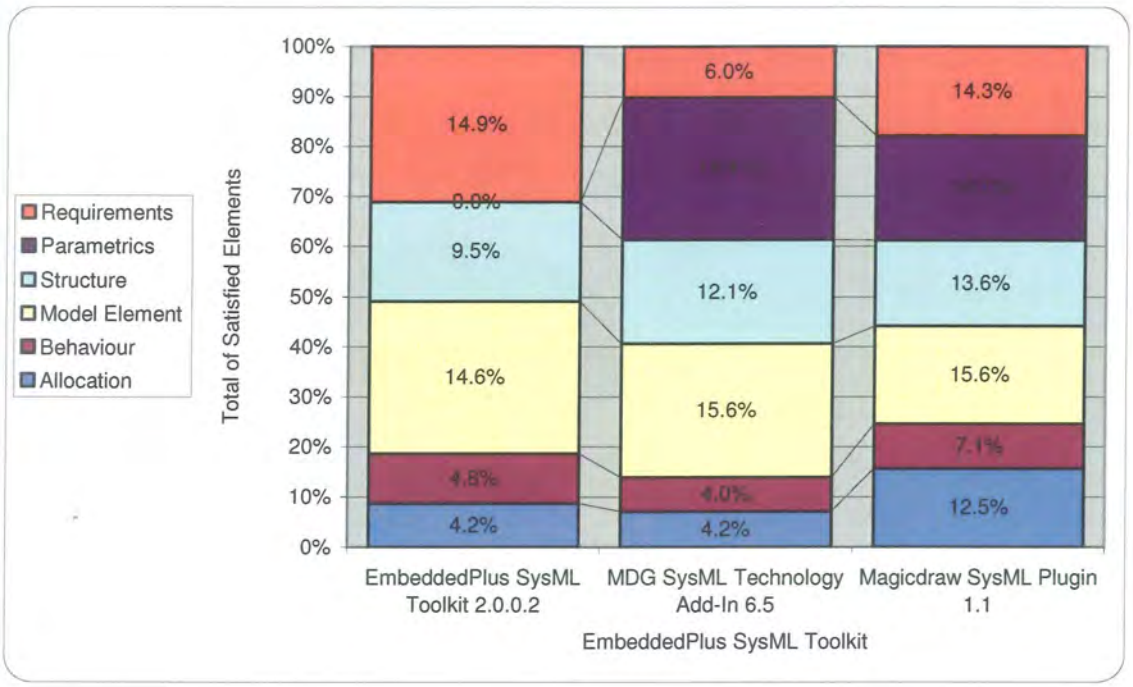


Figure 6.10: Group results using even weighting factors.

Table 6.8 shows that groups with a low quantity of framework elements may be over-represented in this analysis. For instance, the smallest group, "Parametrics", has an

equal weighting factor to "Structure" and "Requirements", two of the largest framework groups.

The next analysis phase considers that the framework groups were not created equal and that each bears a level of significance. It will perform adjustments to the weighting factors for each group and explain their effects on the results.

6.6.4 Fourth Phase using Weighted Ranking with Proportions

This phase will use weighting factors for each group that are based on their proportion to the entire framework. These proportions are relative to the quantity of elements in each of the six groups. Figure 6.11 displays the proportion of each group in relation to the total of framework elements.

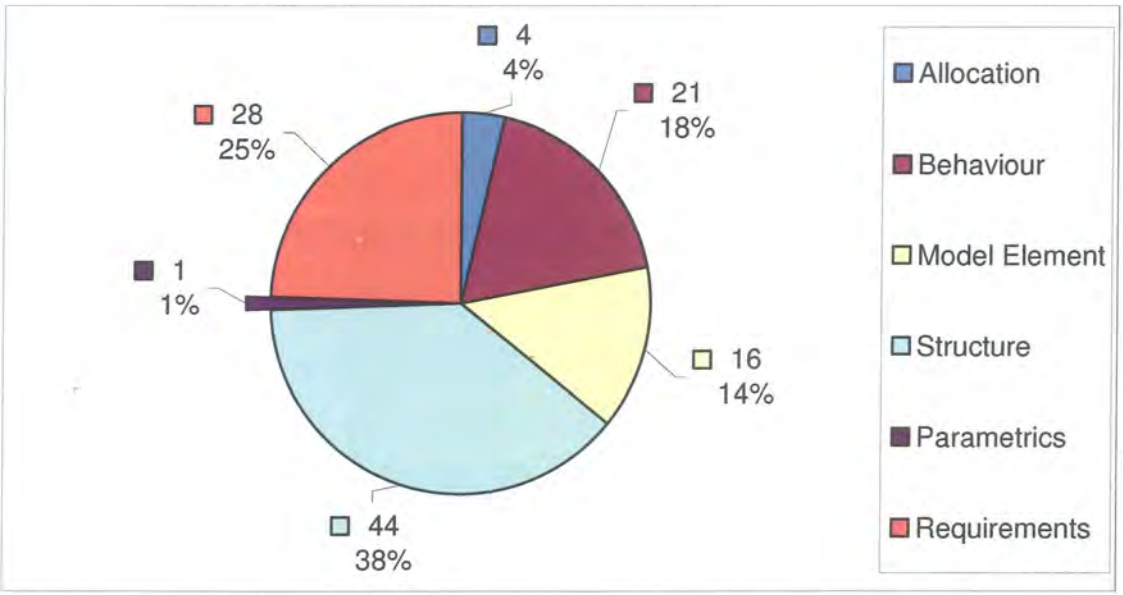


Figure 6.11: Proportion of elements to each framework group.

An assumption may be made about the importance of a group based on how much of the framework is devoted it. For instance, figure 6.11 shows that "Structure" has a greater magnitude than "Model Element", "Behaviour" or "Allocation".

Table 6.9 and the extrapolated results in figure 6.12 show results using weighting factors for groups based on their element proportion.

Table 6.9:

Summary of group results using proportional weighting factors.

		Candidate		
Group	Weighting Factor	EmbeddedPlus SysML Toolkit 2.0.0.2	MDG SysML Technology Add-In 6.5	Magicdraw SysML Plugin 1.1
Allocation	4%	0.9%	0.9%	2.6%
Behaviour	18%	5.3%	4.4%	7.9%
Model Element	14%	12.3%	13.2%	13.2%
Structure	39%	21.9%	28.1%	31.6%
Parametrics	1%	0.0%	0.9%	0.9%
Requirements	25%	21.9%	8.8%	21.1%
	TOTAL	**62.3%**	**56.1%**	**77.2%**

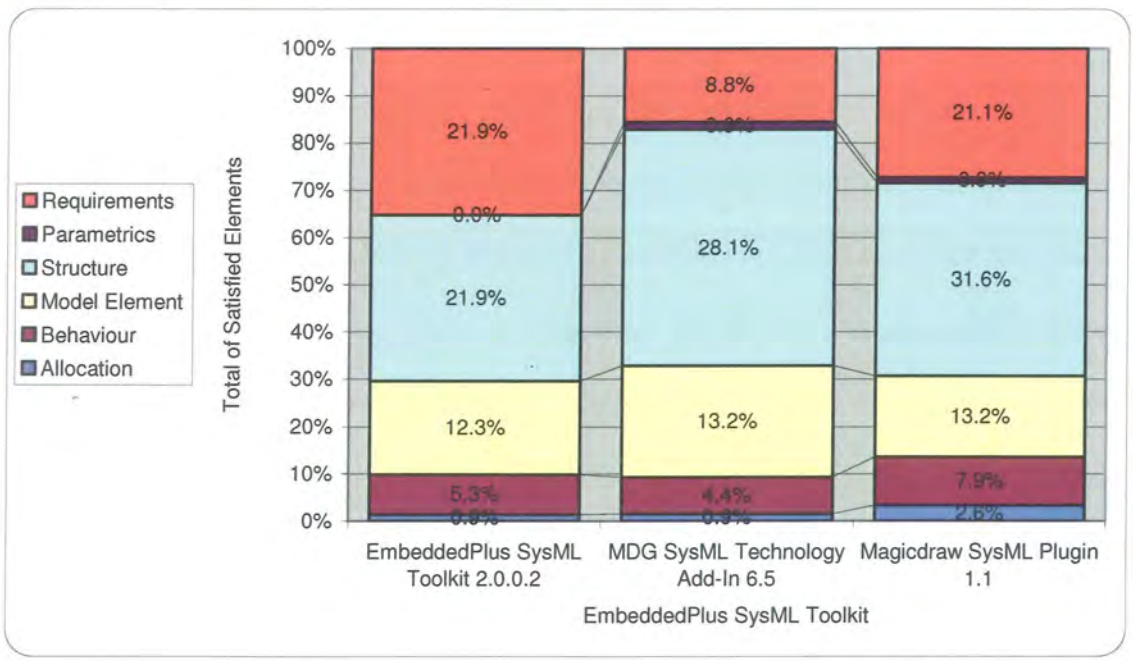


Figure 6.12: Results with Proportional Weighting Factors.

The weighting factor for a group was calculated as a percentage of total framework elements that belong to that group.

A comparison of table 6.8 and table 6.9 reveals that the weighting factors significantly influence the final scores for each candidate. As a result of this adjustment, the EmbeddedPlus tool ranked second. Its final score increased by 14.4 percent. Also, it was influenced the most since the Sparx Systems and Magicdraw tools decreased in final score by 2.4 and 2.7 percent, respectively.

In this analysis, the groups with the greatest number of elements influence the candidate rankings the most. The Magicdraw tool's scores in the largest groups, "Structure", "Behaviour" and "Requirements", were major factors in its ranking.

In this phase, a group's influence on a candidate's final score was determined by its size within the framework. A deeper results analysis could have considered the importance of each element in relation to their allocated group. For instance, the FAS's description of the "Block" metaclass reads: "SysML blocks can be used throughout all phases of system specification and design, and can be applied to many different kinds of systems" (OMG, 2006, p. 33). "Block" could be considered an important part of the structural modelling aspects of the SysML's. Therefore, the framework elements that address the "Block" metaclass could be given a weighting factor as a measurement of their importance to the "Structure" group. This weighting factor would be relative to the total number of elements within the group. Due to the immensity of the framework and time constraints, this analysis direction was not considered.

The results in table 6.9 show the amount of elements from each weighted grouping that were satisfied by a candidate. The weighting factor for a grouping is based on the number of elements contained within that particular grouping. In comparison to the analysis results shown in table 6.8, this approach shows the significance of each group based on their weighting factor as well as a candidate's mark for each category. However, this approach merely provides a different perspective of the results. The

results for each category and the final scores for each candidate are still proportional to those obtained using the previous analysis approach.

6.6.5 Fifth Phase using Weighted Ranking with Significance

This phase will adjust the weighting factors for each group based on assumptions about their importance to applications in systems engineering. Several studies (Friedenthal, Moore, & Steiner, 2006; Jansma & Jones, 2006; Y. Vanderperren & Dehaene, 2005b) describing systems engineering methods will be used to base these assumptions.

Kayton (1997) and the Institute of Electrical and Electronics Engineers (IEEE) (2000) would agree that there are numerous systems engineering methods. However, these methods appear to have similarities. For example, Friedenthal, Moore, & Steiner (2006), Vanderperren & Dehaene (2005b) and Jansma & Jones (2006) all mention "requirements definition" as being a significant part of their systems engineering process.

The OOSEM, which was developed in collaboration with Lockheed Martin Corporation and the Software Productivity Consortium, is explained by Lynkins, Friedental, and Meilich (2000) and Estefan (2007). It was intended to address the needs of systems engineering using methods from software engineering, such as the use of OO models. The system development activities of this method are highlighted by Friedenthal, Moore, & Steiner (2006) and consist of: "Analyse Needs", "Define System Requirements", "Define Logical Architecture" and "Synthesise Physical Architecture".

In their discussion of the SysML in relation to SoC design, Vanderperren & Dehaene (2005b) mention the systems engineering process: "SIMILAR". According to Bahill & Dean (2007), SIMILAR is an iterative process that incorporates a system development life cycle. This life cycle consists of activities for requirements discovery; investigation

of alternative designs; design of the entire system; system implementation; component integration and integration testing; maintenance, operation and performance evaluation; and system retirement. Determining the customer's needs, in order for requirements to be specified and validated, is of the upmost importance to this systems engineering process.

Vanderperren & Dehaene consider the importance of the "requirements engineering" process and mention that the requirements modelling diagram in SysML can assist in this process. Vanderperren & Dehaene also consider modelling the SysML's "ViewPoint" element to be of particular importance to requirements validation and mentions examples its application to SoC projects.

In Jansma & Jones (2006), research was conducted by a team of the SEA project for improving systems engineering practices at Jet Propulsion Laboratory (JPL). They identified functions covering system architecture, requirements and interface definition; resource coordination; validation and verification of requirements; risk engineering; technical reviews; and management of the design and systems engineering processes.

Kayton (1997) elaborates on a definition of a systems engineering process that focuses on system design. It approach for system design consists of "translating" the needs of the customer, defining subsystem interfaces, performing risk management and verifying the system design against it's specified requirements. Kayton states the importance of systems engineers within projects and mentions, amongst other major systems engineering responsibilities, requirements analysis and the task of integrating and assembling subsystems.

The "Tactical Science Solutions" team of George Mason University performed an evaluation of the SysML to assess its suitability to MBSE. The team developed an iterative, "hierarchical design method" to be used in conjunction with the SysML.

Each cycle of this method focuses on performing behaviour and requirements analysis for a single high-level system block. The behaviour analysis leads into structural definition for the block's lower levels using internal block and block definition diagrams. Behaviour definition is then applied to the lower-level structures using either state machine or activity diagrams. This is then followed by a confirmation that the requirements have been satisfied, documentation and modelling using parametric diagrams for supporting executable models.

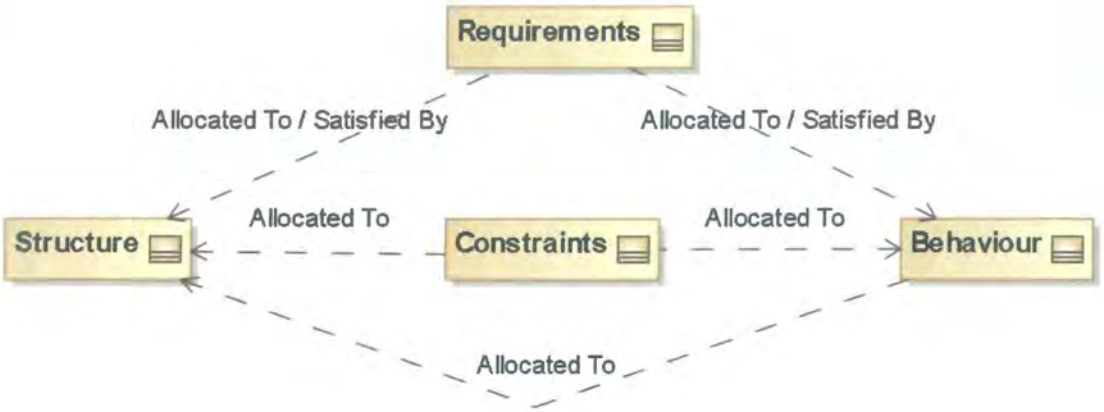
In this process, requirements are used to drive the high-level analysis stage and confirm the lower-level logical decomposition; requirements appear to be a significant aspect of this process. "Functional analysis" and "logical analysis" lead onto behavioural and structural definition, respectively.

According to OMG, "SysML blocks can be used throughout all phases of system specification and design, and can be applied to many different kinds of systems." (OMG, 2006, p. 33). OMG identifies the versatility of the Block metaclass; it is an essential ingredient to structural modelling within the SysML. When comparing the significance of this item to other allotted items in the "structural" aspect of the evaluation framework, a higher weighting rank may be afforded to the more significant item.

"Activity" is another significant metaclass contained in the "Behaviour" category. It is more fundamental to behavioural modelling than other elements within that category,

such as the "Optional" stereotype or the "Rate" stereotype, and therefore could be afforded a higher significance value than those other elements.

Colombo et al. (2006) illustrate an approach that uses the SysML to perform systems modelling and this is reproduced in figure 6.13.



(Colombo et al., 2006)

Figure 6.13: Modelling approach used by Colombo et al.

Figure 6.13 shows the relationships of "Behaviour", "Structure" and "Constraints" modelling in respect to "Requirements".

Balmelli (2006) describes the importance of using system models to define context and goes on to describe how system context is defined using SysML block diagrams.

A common set of processes can be identified within these studies, namely:

1. Requirements Definition;
2. Structural Design;
3. Functional Decomposition;
4. Interface Definition;
5. Implementation; and

6. Evaluation and Analysis;

The "UML for Systems Engineering" RFP (refer to section 2.3) defines several requirements for a "general purpose systems modelling language" and these are addressed by the SysML (OMG, 2003, p. 23). These requirements consider the modelling of a system's structure, behaviour, requirements and internal properties (including parametric equations) as well as behaviour and requirements allocation.

The RFP's stated requirements appear to be consistent with the process defined by Bahill & Dean (2007). For instance, the RFP stipulates the ability to define system structure and perform functional decomposition. Kayton (1997) mentions that systems engineers partition a system's structure into subsystems and Bahill & Dean (2007) mention design activities that require this. Figure 6.14 shows a concept map of the major systems engineering concepts mentioned by Bahill & Dean and the RFP (OMG, 2003).

These studies emphasise the importance of requirements to the engineering process for defining, analysing and verifying structure and behaviour. Therefore, an assumption can be made about the significance of the "Requirements" group containing elements for designing requirements definitions and relationships. Given the discussions on systems engineering mentioned so far it is not surprising that, in descending order of size, "Structure", "Requirements" and "Behaviour" are the largest groups of elements gathered for the framework from the FAS.

The FAS explains which parts of the SysML 1.0a address specific parts of the "UML for Systems Engineering" RFP (OMG, 2006, p. 223).

Peak et al. (2007a; 2007b) successfully applied the SysML's analysis, structure, behaviour and requirements modelling capabilities to a "simulation-based design"

project. This, and other studies applying (Colombo et al., 2006) and discussing (Vanderperren & Dehaene, 2005a; Vanderperren & Dehaene, 2005b; Viehl et al., 2006; Yves & Wim, 2006) its capabilities are indicative of the SysML's applicability to systems engineering.

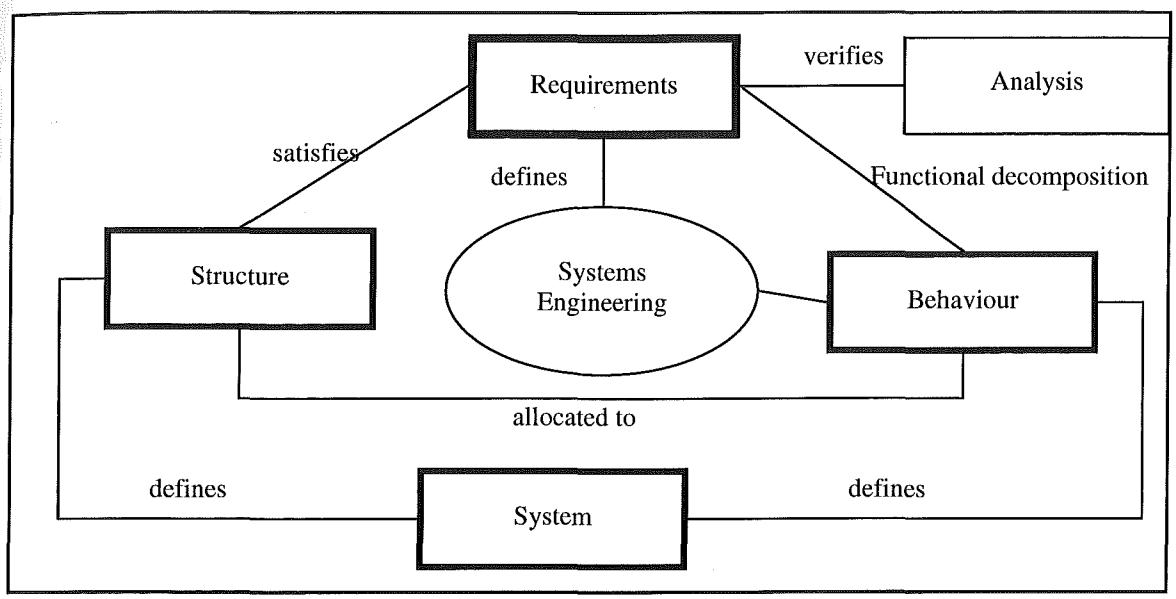


Figure 6.14: Systems Engineering concepts (Bahill & Dean, 2007; OMG, 2003).

Friedenthal et al. (2006) illustrates the relationships between the four modelling aspects of SysML. "Cross-cutting" elements facilitate these relationships. The illustration explains that "Behaviour" elements are allocated to "Structure" elements, which are subject to property constraints from "Parametrics" elements. "Requirements" elements are satisfied by "Structure" elements and are verified by "Parametrics" elements.

This analysis will set the weighting factors for each group based on the following assumptions.

1. Requirements are the most important aspect of the development process since they are input to a number of activities in systems engineering;
2. Structural and behavioural definitions are developed to satisfy requirements and behaviour is allocated to structure;

3. Allocations are essential in SysML for associating separated system structure and behaviour models with each other and associating physical (OMG, 2006). "Structure allocation is associated with the concept of separate "logical" and "physical" representations of a system." (OMG, 2006, p. 127); and
4. Parametrics are used to constrain properties and model relationships between properties using constraints, mathematical equations and logical expressions (OMG, 2003, p. 37). They allow architectural and requirements models to be associated with analysis models by "binding" specific system properties to the parameters of engineering constraints (Peak et al., 2007a).

When considering the results from the previous analysis, the adjustments have decreased the final scores for the EmbeddedPlus tool and the Sparx Systems tool by 7.2 percent and 4.2 percent, respectively. The final score for the Magicdraw tool received only a slight change since, with the exception of the "Requirements" group, the tool satisfies the most group elements overall. The EmbeddedPlus tool received the highest score for the "Requirements" group.

When comparing the group results of the closely matched EmbeddedPlus and Sparx Systems tools, the EmbeddedPlus tool lead in "Behaviour" and "Requirements", whilst the Sparx Systems tool lead in "Model Element", "Structure" and "Parametrics".

Table 6.10:

Summary of group results using adjusted weighting factors.

Weighting Factor	Group	Candidate: EmbeddedPlus SysML Toolkit 2.0.0.2	Candidate: MDG SysML Technology Add-In 6.5	Candidate: Magicdraw SysML Plugin 1.1
10%	Allocation	2.5%	2.5%	7.5%
20%	Behaviour	5.7%	4.8%	8.6%
10%	Model Element	8.8%	9.4%	9.4%
20%	Structure	11.4%	14.5%	16.4%
10%	Parametrics	0.0%	10.0%	10.0%
30%	Requirements	26.8%	10.7%	25.7%
	TOTAL	**55.1%**	**51.9%**	**77.5%**

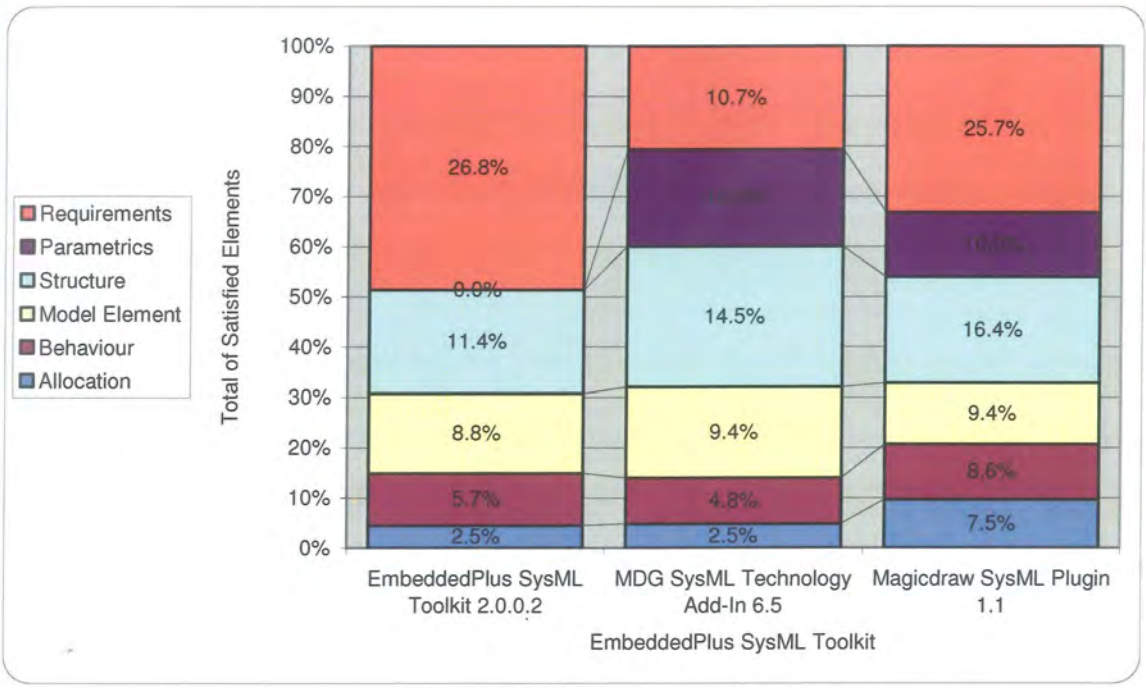


Figure 6.15: Results using adjusted weighting factors.

7 Qualitative Evaluation

This section will present an analysis of the qualitative evaluation results using several software quality engineering factors described by Deutsch and Willis (1989). These factors will apply a structure to the analysis and, where appropriate, certain software quality terms will be defined.

The evaluation results were gathered during the modelling of the HSUV sample problem that is available from the SysML FAS document. An example of this problem is provided by the vendors of each candidate tool in the form of a modelling project. However, some of these examples were incomplete. To ensure that each candidate could be assessed equally on each part of the problem, further development of these examples was required.

The EmbeddedPlus tool contained the least complete sample model and required the most development work for the evaluation. In terms of usability, the EmbeddedPlus tool proved the most difficult candidate to evaluate due to program faults with its SysML extension.

The raw validation results are available in appendix d: qualitative evaluation results. The results indicated that, with the Magicdraw tool and the Sparx Systems tool, the sample model triggered rules concerned with "ObjectFlow" and "Requirement" elements. The majority of the results for the EmbeddedPlus tool indicated that rules concerned with the "Connector" element were triggered.

The evaluation found that the candidates were able to represent most of the sample model. As in the quantitative evaluation, only the figures from the FAS showing Requirements, Parametrics, Internal Block, Block Definition and Activity diagrams

were considered in this evaluation. A few figures within the FAS were not considered due to their dependence on language elements that were not part of the standard SysML. An example would be the figure in the FAS labelled "Detailed Behavior Model for 'Provide Power'" (OMG, 2006, p. 202), which contains an activity diagram using non-standard SysML notation.

A few observations were made during the evaluation. For example, a Parametrics diagram titled "Establishing Mathematical Relationships for Fuel Economy Calculations" from the FAS's sample problem clearly shows references to nested value properties located within the model, as indicated by their names. However, in the Sparx Systems tool's sample model, the value properties have been made to appear as if they are nested value properties. A screenshot is provided in figure 7.1 to elaborate on this.

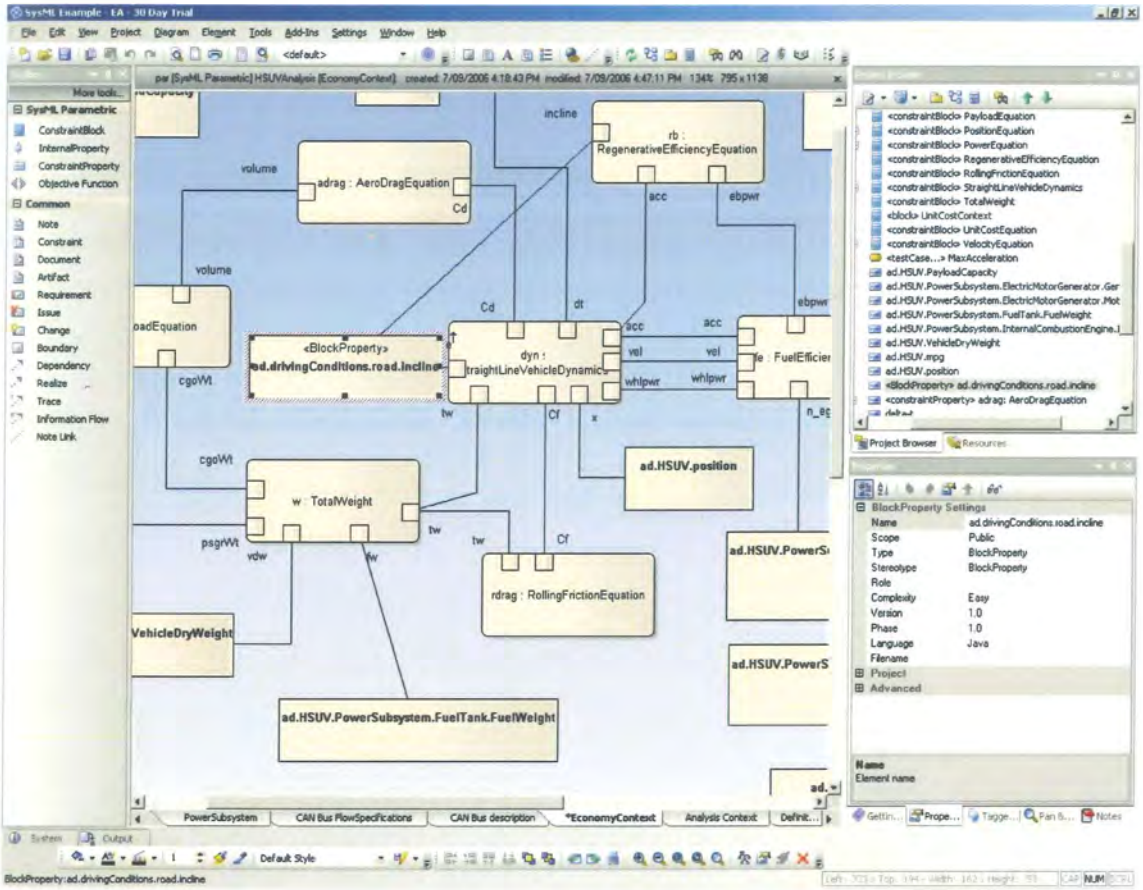


Figure 7.1: Sparx Systems tool's sample model showing "nested" value properies.

The evaluation also found that certain tools were unable to completely represent the sample model from the FAS. Figure 7.2 is an internal block diagram from this model. It shows "item flows" on connectors between flow ports, which are represented as solid arrow heads labelled with a name and a type definition. For example, "fuelSupply:Fuel" indicates an item flow named "fuelSupply" that is defined with the type "Fuel".

The EmbeddedPlus tool was unable to represent this part of the model properly since it does not support "item flows" on connectors (see figure 6.6). This lack of support may be a factor in its incomplete implementation of the sample problem, which was intended to demonstrate the SysML's fundamental features. It appears unlikely that the EmbeddedPlus tool could accommodate modelling problems that require this unsupported feature.

Also, certain "non-normative" extensions were required by the FAS sample problem and were not implemented in the EmbeddedPlus tool. These include the "measure of effectiveness" («moe») and "objective function" («objectiveFunction») stereotypes that are used by certain parts of the sample problem (OMG, 2006).

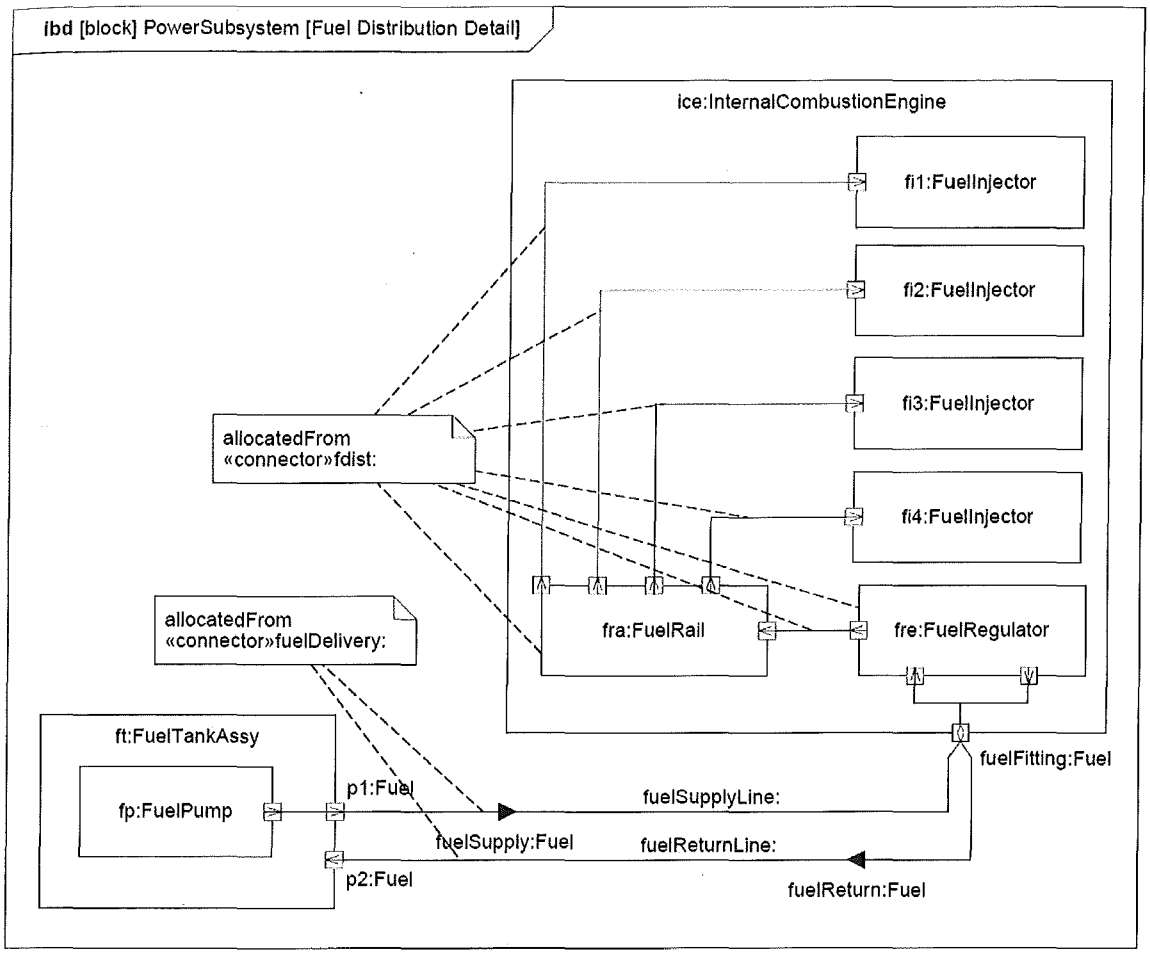


Figure 7.2: "Detailed Internal Structure of Fuel Delivery Subsystem"(OMG, 2006, p. 192)

This evaluation acknowledges that modelling tool vendors may interpret the FAS differently and therefore may implement the language's rules and syntax differently. Also, the SysML is an evolving language and tool vendors may choose to implement different versions of its specification. For example, figure 7.3 is a screen capture from the Sparx Systems tool that shows the version information for its SysML extension. It states that the tool implements the OMG SysML Draft Adopted Specification (DAS) - a specification earlier than the FAS.

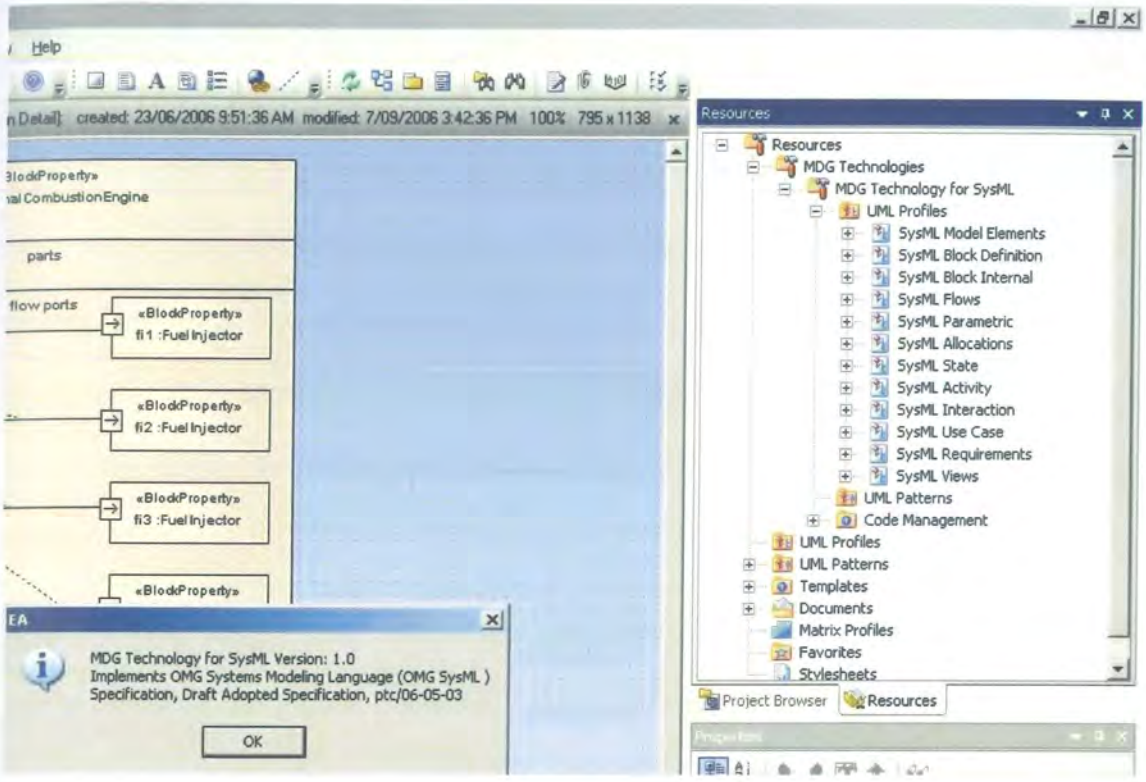


Figure 7.3: Implementation details of the SysML extension for the Sparx Systems tool.

The different responses obtained from the validation functions of each candidate may be a consequence of the developer's implementation. Juric & Kuljis (1999) mention that tools may be targeted to the modelling preferences of certain user groups. Juric & Kuljis evaluated several CASE tools for the UML. In their results discussion, they mention that the behaviour of a tool's validation mechanism may determine its suitability to individuals with different UML experience levels. The development tool preferences of experienced UML modellers may differ from those who are less-experienced. An experienced modeller may desire a tool with more dynamic functions, fewer restrictions on how diagrams are composed, fewer boundaries to the development process and a higher prior knowledge expectation than that of a tool preferred by a UML novice (Juric & Kuljis, 1999, p. 9). During their investigation, Juric & Kuljis discovered the need for the UML creators to offer compliance guidelines in order for an investigator to determine the extent to which a CASE tool should incorporate the UML.

The candidate tools used within this research have different approaches for enabling end-users to modify how constraints are enforced and how feedback is communicated to them. For example, the EmbeddedPlus tool allows users to selectively enable or disable constraints for a particular modelling notation, such as the SysML. Each candidate tool allows the user to choose between different sets of model validation rules for various languages. This capability may allow end-users to establish preferences for the modelling environment that suit their own knowledge of the language syntax and semantics.

If an idealised model was incorporated into future research, it could be designed to address each of the groups contained within the comparative framework. The model could be used as a reference model for evaluating different modelling aspects of the FAS. If this reference model could be represented as an XMI source file, it could be used to evaluate modelling tools that are capable of interpreting that data format. Language compliance could then be measured by assessing the modelling tool's ability to represent the reference model.

The next section will focus on the software quality aspects of the candidates.

7.1 Analysis

This section contains the results of stage five and six of the interpretive research process shown in figure 3.3. It reflects on the modelling experience received from applying the HSUV sample model to the candidate tools.

There are disadvantages to incorporating the sample model from the SysML FAS into this evaluation. One disadvantage is that it consists entirely of annotated diagrams. It lacks a complete textual specification for the HSUV system and a preamble to its development. Such information may exist in a more realistic engineering problem and

benefit the quality of the evaluation. Another disadvantage is that it requires an existing library of SI Unit and Dimension definitions (OMG, 2006, p. 211). An implementation of this library existed in the Magicdraw tool and the Sparx Systems tool.

In the regards to its advantages, as mentioned in the FAS (OMG, 2006), it provides a way to demonstrate the basic functionality of the SysML. As an ideal model, the sample problem provides an adequate benchmark for evaluating candidates.

Deutsch and Willis (1989) describe factors that contribute to software quality and their advice was incorporated into this evaluation in order to enhance its results. They define fifteen quality factors that focus on a user's needs and can be used to engineer the quality of a software product. Out of these factors, efficiency, reliability, usability, correctness, flexibility and verifiability were selected to guide this evaluation. They were chosen since this evaluation does not focus on quality factors such as software interoperability, expandability, robustness, safety considerations or compatibility with system architectures. Also, in accordance with the research design in section 3.1, the interoperability of the candidates and the portability of the sample model between them were not considered.

7.1.1 Correctness

According to Deutsch and Willis (1989), correctness refers to how well a software product satisfies its initial design. Due to the unavailability of software design artefacts for each candidate, this section will consider user documentation as a substitute.

User documentation accompanied each candidate tool. The documentation from the Sparx Systems tool was unique in that it elaborated on its SysML modelling features and matched them to relevant sections of the SysML language specification. The

EmbeddedPlus tool was the only candidate to contain errata in its documentation, which listed the software's incomplete features and unsupported language elements.

The candidates satisfied the modelling features described by their user documentation. With the Sparx Systems tool, an issue was discovered with one particular element, the enumeration element: "ControlValue". According to the OMG (2006), if the «ControlOperator» element is applied to an operation feature of a Block or Activity, a minimum of one parameter within that operation is required to be typed by the ControlValue element. This element is not available in the Sparx Systems tool's SysML profile. Instead, the user is required to create an element named "ControlValue" and use that element in order to satisfy the tool's model validation.

7.1.2 Efficiency

In this section, efficiency will be measured in terms of a candidate's responsiveness and modelling performance during the evaluation. The EmbeddedPlus candidate was the worst in terms of efficiency. It provided the longest waiting times for loading diagrams and performing modelling functions. The Sparx Systems tool performed best in this category.

7.1.3 Flexibility

To ensure the relevancy of this section to the evaluation, the ability of each tool to provide flexibility in terms of UML profiles will be considered. Each candidate is capable of integrating several UML profiles into their user environment.

During the evaluation, the Magicdraw tool was found to provide the most flexibility for handling UML profiles. The tool permits introspection of the properties and relationships of elements within profiles referenced by the currently loaded modelling

project file. For example, a user is able to select a stereotype element within one of a library of profiles, such as the SysML profile. The user may then view its features, use it to compose a diagram or display its relationship to other elements. This capability was a particularly importance source of empirical evidence for both the comparative and qualitative evaluation. It was unmatched in terms of versatility by the remaining candidates.

Profiles may be mastered and viewed within the EmbeddedPlus tool. However, the tool appears to prevent profile elements, such as metaclasses, from interacting with content from a user project, such as diagrams and packages user. Also, the tool environment does not inform the user of which profiles are loaded for an active modelling project.

Figure 7.3 shows the Sparx Systems tool's representation of its SysML profile. Apart from the introspective capabilities of this feature, it does not offer the level of detail available in the Magicdraw or EmbeddedPlus tools.

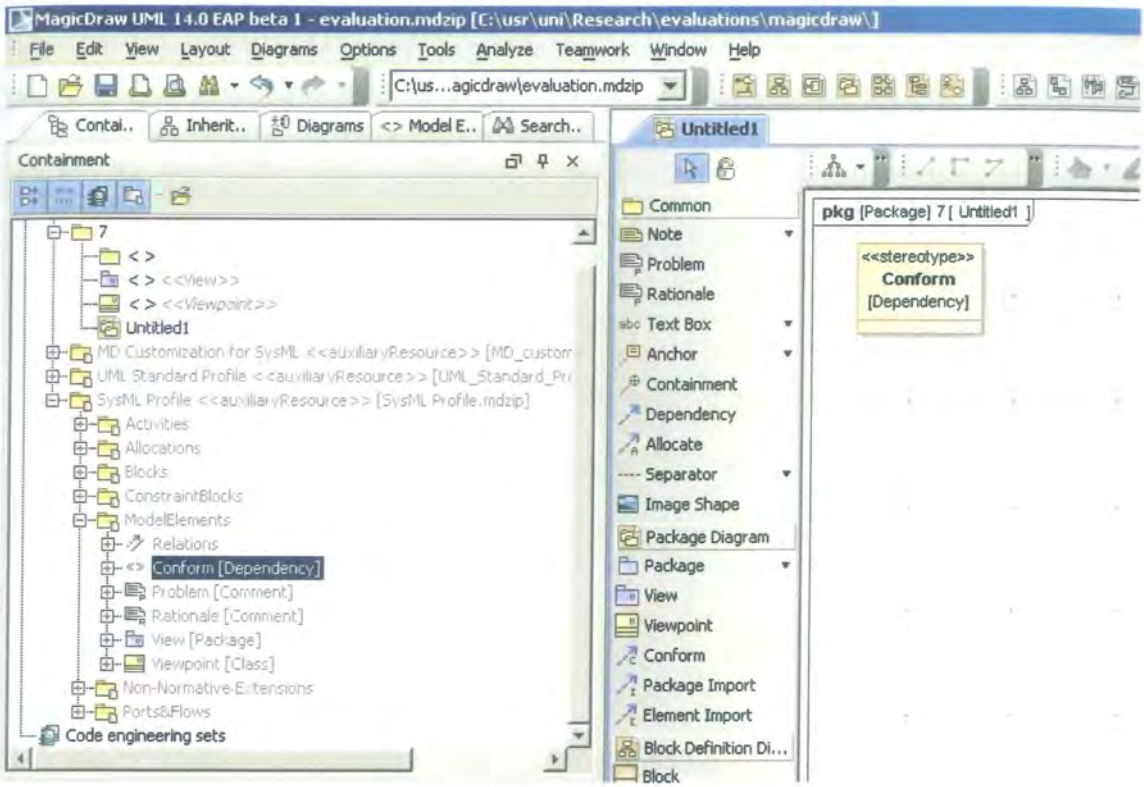


Figure 7.4: A view of a SysML profile's contents in the Magicdraw tool.

7.1.4 Usability

Deutsch and Willis state that usability "deals with the initial effort required to learn, and the recurring effort to use, the functionality of the software" (1989, p. 49). The usability of the Sparx Systems tool is affected by the level of information that it communicates to the end user when developing a model. An example of this is the visual representation of relationships. The EmbeddedPlus and MagicDraw tools both provide informative, editable representations of relationships, such as connectors between ports, within the view of a SysML model's hierarchy of content. In the Sparx Systems tool, the "Project Browser" is restricted to showing only UML packages, elements and their features. However, relationships can be observed on a per element basis by viewing the elements' properties dialog box. Also, features such as the "Relationship Matrix" tool are available for querying relationships within a model.

Each candidate provided a different approach to structural modelling using Internal Block Diagrams and Block Definition Diagrams. For example, the EmbeddedPlus tool renders the internal features of a Block differently to its peers. It permits elements to be defined within the structure compartment of a Block. By default, the tool preserves the positions and layouts of all ports, parts and connectors within the structure compartment of a Block. If the Block is viewed in a separate diagram, this preservation of configuration may present a difficulty to those wishing to view a subset of that internal structure.

7.1.5 Reliability

Reliability is defined by Deutsch and Willis as dealing with "the rate of failures in the software that render it unusable" (1989, p. 49). The Sparx Systems tool demonstrated the worst reliability during the evaluation. An example of its instability exists with the "Embedded Elements" dialog box, which allows the user to add a new part to a Block. This function would occasionally cause the application to immediately terminate. An estimation of the frequency of these failures would be approximately one every 24 hours.

Certain issues of the EmbeddedPlus tool ensured its ranking as the second-most reliable candidate. These included issues with rendering the internal features of elements, such as Blocks and Activities, and problems with loading diagrams. Figure 7.5 shows EmbeddedPlus failing to load an existing diagram.

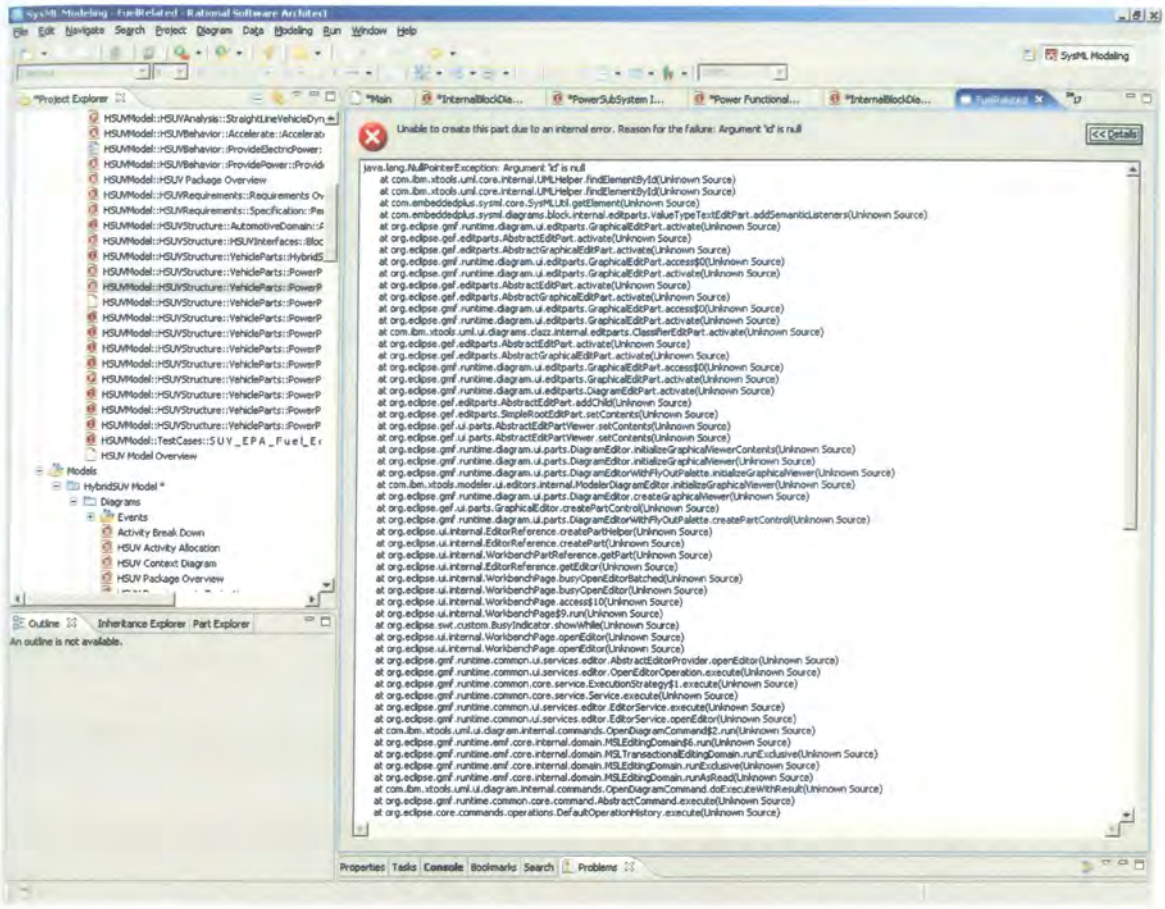


Figure 7.5: The EmbeddedPlus tool's failure to load an existing diagram from the sample model.

7.1.6 Verifiability

According to Deutsch and Willis, verifiability is defined as "how easy it is to verify that the software is working correctly" (1989, p. 49). This section will consider verifiability in terms of model validation.

Each tool provides facilities for validating an actively loaded SysML model. The validation mechanisms provide a tabular view for the results of a recent validation execution. Also, informative messages may appear if the end-user performs an action that may invalidate the model within the current diagram view.

The candidates provided adequate model validation functions for the evaluation.

8 Conclusion

This research was designed to answer the following research questions:

1. To what extent do current SysML modelling tools implement key parts of the OMG SysML 1.0 language specification?
2. Can a model of a physical, real-world system be represented consistently between the current SysML modelling tools?

In regards to the first question, the comparative evaluation found that the candidates varied in compliance to the element framework. Overall, the Magicdraw tool received the highest compliance score throughout the results analysis.

The analysis with even weighting factors showed that, in terms of satisfied elements within each framework group, the Sparx Systems tool performed better than the EmbeddedPlus tool (see table 6.8).

Overall, the candidates did not satisfy more than 42.9% of the framework's behavioural elements. The results of the comparative evaluation indicate that certain candidate tools may be more compliant with a particular framework group than other candidates. This may be the effect of software vendors aligning tools with the modelling needs of certain users, which is described by Juric & Kuljis (1999). For example, in the case of modelling system behaviour, the comparative evaluation found that the Magicdraw tool satisfied the most framework elements for behavioural modelling. As another example, when comparing the EmbeddedPlus tool with the Sparx Systems tool, the former appears to satisfy more elements for requirements modelling and the latter appears to satisfy more elements for structural modelling.

The final analysis results in table 6.10 indicate that the Magicdraw tool satisfies the most elements that are significant to systems engineering.

In regards to the second question, the results from the qualitative evaluation were inconclusive. A conclusive result could not be obtained with the chosen independent variable, which consisted of the output of each candidate's validation mechanism. However, the validation results were able to identify areas of the sample model that were incomplete.

The time constraints of this research were sufficient for evaluating the abstract language syntax for the SysML extensions to the UML. This evaluation could be further enhanced by also considering compliance with a candidate's underlying UML implementation.

9 Future Work

In regards to future research related to this research, future versions of the SysML specification could be subject to similar comparative and descriptive or interpretive evaluations. For example, the availability of the OMG SysML 1.0 Available Specification (OMG, 2007b) may compel researchers to consider that specification in their assessments.

This research concentrated on the SysML extensions to the UML. A more thorough investigation of modelling tool compliance could consider both the UML 2.1 and the SysML 1.0 specifications.

Future evaluations may consider other forms of language syntax when assessing the compliance of modelling tools. Compliance with concrete syntax, such as notational features and diagram appearance, and abstract syntax, such as the interpretation of XMI sources, could be considered (OMG, 2006, p. 15). An evaluation that considers XMI could determine if reliable interchange and preservation of model information is possible using the current modelling tools. Also, the interchange of diagram information (OMG, 2006, p. 170) between modelling tools is another consideration requiring further and more in-depth research.

10 References

Bahill, T., & Dean, F. (2007). "What Is Systems Engineering? A Consensus of Senior Systems Engineers". Retrieved October 4, 2007, from http://www.sie.arizona.edu/sysengr/whatis/whatis.html

Bahill, T., & Gissing, B. (1998). Re-evaluating systems engineering concepts using systems thinking. *IEEE Transactions on Systems, Man and Cybernetics, Part C, 28*(4), 516-527.

Balmelli, L., Brown, D., Cantor, M., & Mott, M. (2006). Model-driven systems development. *IBM Systems Journal, 45*(3), 569-585.

Bar-Yam, Y. (2003). When systems engineering fails-toward complex systems engineering. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (pp. 2021-2028). Washington: New England Complex Systems Institute.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly, 11*(3), 369-386.

Bock, C. (2005). Systems engineering in the product lifecycle. *International Journal of Product Development, 2*(1), 123-137.

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (Second ed.). New Jersey: Addison-Wesley.

Burks, H. L. (1991). Systems engineering Tools. *Proceedings of the Reliability and Maintainability Computer-Aided Engineering in Concurrent Engineering Workshop* (pp. 241-244). Leesburg: Texas Instruments.

Colombo, P., Bianco, V. D., Lavazza, L., & Coen-Porisini, A. (2006). *An Experience in modeling real-time systems with SysML* Paper presented at the 9th International Conference on Model Driven Engineering Languages and Systems, Genova. Retrieved April 12, 2007 from http://www.martes.org/prev/2006/SLIDES/12.pdf.

Denno, P. (2006). "A Systems Engineering Tool Interoperability Plug-fest". Retrieved April, 20, 2007, from http://syseng.nist.gov/se-interop/static/anaheim-2006-09-27.pdf

Deutsch, M.S., Willis, R.R. (1988). *Software Quality Engineering: A Total Technical and Management Approach.* New Jersey: Prentice-Hall.

"EmbeddedPlus SysML Toolkit for the IBM RSDP". (2007). Retrieved October 5, 2007, from http://www.embeddedplus.com/downloads/SysML%20Toolkit%20for%20RSDP.pdf

Estefan, J. A. (2007). *Survey of Model-Based Systems Engineering (MBSE) Methodologies.* Pasadena: Jet Propulsion Laboratory, California Institute of Technology.

Floyd, C. (1986). A comparative evaluation of system development methods. *Proceedings of the IFIP WG 8.1 working conference on Information systems design methodologies: improving the practice* (pp. 19-54).

Friedenthal, S., Moore, A., & Steiner, F. (2006). OMG Systems Modeling Language (OMG SysML) Tutorial. *INCOSE International Symposium,* from http://www.omgsysml.org/SysML-Tutorial-Baseline-to-INCOSE-060524-low_res.pdf

Galliers, R. D. (1990). Choosing Appropriate Information Systems Research Approaches: A Revised Taxonomy. In H.-E. Nissen, Klein, H.K. and Hirschheim-Stuart, R., (Ed.), *The Information Systems Research Arena of the 90's: Challenges, Perceptions and Alternative Approaches, Proceedings of IFIP TC8 WG8.2 Conference* (pp. 155-173). Copenhagen, Denmark.

Ganesan, S., & Prevostini, M. (2006). *Bridging the Gap between SysML and Design Space Exploration.* Paper presented at the Forum on Specification & Design Languages '06, Darmstadt, Germany. Retrieved April 15, 2007.

Goering, R. (2006). System-level design language arrives (SysML) [Electronic Version]. *Electronic Engineering Times.* Retrieved April 15, 2007 from http://www.eetimes.com/issue/fp/showArticle.jhtml?articleID=187200782

Hause, M., Thom, F., & Moore, A. (2004). The Systems Modelling Language-SysML. Retrieved April 15, 2007, from http://www.omgsysml.org/ARTiSAN-The_Systems_Modeling_Language.pdf

Hause, M., Thom, F., & Moore, A. (2005). An overview of Systems Modeling Language. *Embedded Systems Design, 18*(12), 39.

IEEE. (2000, October). What is Systems Engineering? *IEEE Aerospace & Electronic Systems Magazine, 15,* 9-10.

INCOSE. (2006). A Consensus of the INCOSE Fellows. Retrieved October 1, 2007, from http://www.incose.org/practice/fellowsconsensus.aspx

Jansma, P. A., & Jones, R. M. (2006). Advancing the Practice of Systems Engineering at JPL. *IEEE Aerospace Conference*, 19.

Juric, R. (1998). The UML rules. *ACM SIGSOFT Software Engineering Notes, 23*(1), 92-97.

Juric, R., & Kuljis, J. (1999). Building an evaluation instrument for OO CASE tool assessment for Unified Modelling Language support. *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences, 7* (pp. 7040-7050).

Kayton, M. (1997). A practitioner's view of system engineering. *IEEE Transactions on Aerospace and Electronic Systems, 33*(2), 579-586.

Kitchenham, Linkman, S., & Law, D. (1997). DESMET: a methodology for evaluating software engineering methodsand tools. *Computing & Control Engineering Journal, 8*(3), 120-126.

Kobryn, C. (2004). Expert's voice, UML 3.0 and the future of modeling. *Software and Systems Modeling, 3*(1), 4-8.

Kornecki, A. J., & Zalewski, J. (2003). Design Tool Assessment for Safety-Critical Software Development. *Software Engineering Workshop, 2003. Proceedings. 28th Annual NASA Goddard*, 105-113.

Krick, E. V. (1969). *An Introduction to Engineering and Engineering Design* (Second ed.). New York: John Wiley & Son, Inc.

Kühne, T. (2006). Matters of (Meta-) Modeling. *Software and Systems Modeling, 5*(4), 369-385.

Law, D. (1988). *Methods for Comparing Methods: Techniques in Software Development*. Manchester: NCC Publications.

Le Lann, G. (1996). *The Ariane 5 Flight 501 Failure-A Case Study in System Engineering for Computing Systems* (Research Report): INRIA.

LeBlanc, L. A., & Korn, W. M. (1994). A phased approach to the evaluation and selection of CASE tools. *Information and Software Technology, 36*(5), 267-273.

Lundell, B., & Lings, B. (2002). Comments on ISO 14102: the standard for CASE-tool evaluation. *Computer Standards & Interfaces, 24*(5), 381-388.

Lynkins, H., Friedental, S., & Meilich, A. (2000). *Adapting UML for an Object Oriented Systems Engineering Method (OOSEM)*. Paper presented at the 2000 INCOSE Symposium: Software Productivity Consortium, Lockheed Martin Corporation.

"Magicdraw SysML Plugin". (2007). Retrieved October 5, 2007, from http://www.magicdraw.com/files/MagicDraw_SysML_brochure.pdf

McDermid, D. C. (1990). *Software Engineering for Information Systems*. Maidenhead: McGraw-Hill.

McGinnis, L. F., Huang, E., & Wu, K. (2006). Systems engineering and design of high-tech factories. *Proceedings of the 37th conference on Winter simulation* (pp. 1880-1886).

Mueller, W., Rosti, A., Bocchio, S., Riccobene, E., Scandurra, P., Dehaene, W., & Vanderperren, Y. (2006). UML for ESL Design- Basic Principles, Tools, and Applications. *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design '06* (pp. 73-80). San Jose: Paderborn University.

Muth, T. A. (2001). *Modeling Telecom Networks and Systems Architecture: Conceptual Tools and Formal Methods*. New York: Springer.

N.A. (2007). SysML Forum - SysML Tools. Retrieved October 12, 2007, from http://www.sysmlforum.com/tools.htm

OMG. (2001). OMG-Unified Modeling Language, v1.4, Glossary. Retrieved September 20, 2007, from http://www.omg.org/docs/formal/01-09-79.pdf

OMG. (2003). UML for Systems Engineering - Request for Proposal. Retrieved April 30, 2007, from http://www.omg.org/docs/ad/03-03-41.pdf

OMG. (2006). OMG Systems Modeling Language Specification - Final Adopted Specification. Retrieved March 1, 2007, from http://www.sysml.org/docs/specs/OMGSysML-FAS-06-05-04.pdf

OMG. (2007a). Issues for Mailing list of the SysML Finalization Task Force. Retrieved August, 12, 2007, from OMG Issues Website: http://www.omg.org/issues/sysml-ftf.open.html

OMG. (2007b). OMG Systems Modeling Language (OMG SysML™) V1.0 - OMG Available Specification. Retrieved October 1, 2007, from http://www.omg.org/spec/SysML/1.0/PDF

OMG. (2007c). OMG Systems Modeling Language - Official OMG SysML site. Retrieved April 10, 2007, from http://www.omgsysml.org/

OMG. (2007d). Unified Modeling Language: Superstructure, version 2.1.1. Retrieved September 12, 2007, from http://www.omg.org/cgi-bin/apps/doc?formal/07-02-03.pdf

OMG. (2007e). Unified Modeling Language: Infrastructure, version 2.1.1. Retrieved September 12, 2007, from http://www.omg.org/cgi-bin/apps/doc?formal/07-02-04.pdf

Peak, R. S., Burkhart, R. M., Friedenthal, S. A., Wilson, M. W., Bajaj, M., & Kim, I. (2007a). *Simulation-Based Design Using SysML - Part 1: A Parametrics Primer*. Paper presented at the INCOSE International Symposium, San Diego. Retrieved May 4, 2007 from http://eislab.gatech.edu/pubs/conferences/2007-incose-is-1-peak-primer/2007-incose-is-1-peak-primer.pdf.

Peak, R. S., Burkhart, R. M., Friedenthal, S. A., Wilson, M. W., Bajaj, M., & Kim, I. (2007b). *Simulation-Based Design Using SysML - Part 2: Celebrating Diversity by Example*. Paper presented at the INCOSE International Symposium, San Diego. Retrieved May 4, 2007 from http://eislab.gatech.edu/pubs/conferences/2007-incose-is-2-peak-diversity/2007-incose-is-2-peak-diversity.pdf.

Pidcock, W. (2003). What are the differences between a vocabulary, a taxonomy, a thesaurus, an ontology, and a meta-model? , October 5, 2007, from http://www.metamodel.com/article.php?story=20030115211223271

Sage, A. P. (1995). Systems engineering of computer based systems: status and future perspectives. *Proceedings of the 1995 International Symposium and Workshop on Systems Engineering of Computer Based Systems* (pp. 5-15).

Sarle, W. S. (1997). Measurement theory: Frequently asked questions. Retrieved October 5, 2007, from ftp://ftp.sas.com/pub/neural/measurement.html

Selic, B. (2006). UML 2: A model-driven development tool. *IBM Systems Journal, 45*(3), 607-620.

Sibbald, C. (2006). Taming Chaos with SysML. *Software Development, 14*(3), 43, 42 pages.

Skipper, J., Estefan, J., & Shames, P. (2006). NASA/JPL Evaluation Team Position to the INCOSE MDSD SysML Evaluation Process. Retrieved October 12, 2007, from http://groups.google.com/group/SysML-Evaluators/attach/e14d0bef8ae3b991/INCOSE_SysML_Eval_JPL_Input_v1_0-1_10-Jan-2006.pdf?part=4

"Sparx Systems - MDG Technology for SysML". (2007). Retrieved October 5, 2007, from http://www.sparxsystems.com.au/products/mdg_sysml.html

SST. (2005). INCOSE Evaluation: Systems Modeling Language. from http://www.pslm.gatech.edu/topics/sysml/incose-evaluation-2005-12/SST-SysML-for-INCOSE-Evaluation-051219-sf-reva.ppt

Vanderperren, Y., & Dehaene, W. (2005a). UML 2 and SysML: An Approach to Deal with Complexity in SoC/NoC Design. *Proceedings of the conference on Design, Automation and Test in Europe-Volume 2* (pp. 716-717).

Vanderperren, Y., & Dehaene, W. (2005b). *SysML and Systems Engineering Applied to UML-Based SoC Design.* Paper presented at the 2nd UML for SoC Design Workshop, 42nd DAC .

Vessey, I., Jarvenpaa, S. L., & Tractinsky, N. (1992). Evaluation of vendor products: CASE tools as methodology companions. *Communications of the ACM, 35*(4), 90-105.

Viehl, A., Schönwald, T., Bringmann, O., & Rosenstiel, W. (2006). Formal performance analysis and simulation of UML/SysML models for ESL design. *Proceedings of the conference on Design, automation and test in Europe: Proceedings* (pp. 242-247).

Wang, S., Birla, S. K., & Neema, S. (2006). A modeling language for vehicle motion control behavioral specification. *Proceedings of the 2006 international workshop on Software engineering for automotive systems* (pp. 53-60).

Willard, B. (2007). UML for systems engineering. *Computer Standards & Interfaces, 29*(1), 69-81.

Wymore, A. W. (1993). *Model-Based Systems Engineering*. Boca Raton: CRC Press.

Yin, R. K. (1981). The Case Study Crisis: Some Answers. *Administrative Science Quarterly, 26*(1), 58-65.

Yves, V., & Wim, D. (2006). From UML/SysML to Matlab/Simulink: current state and future perspectives. *Proceedings of the Design, Automation and Test in Europe Conference '06* (pp. 1-1). Munich, Germany: European Design and Automation Association.

11 Appendix A – Screening Rationale

11.1 Description

The underlying principle using the FAS to prepare an evaluation framework is to evaluate the implementation of extensions to the UML that are specific to the SysML language. These extensions consist of attributes or constraints defined as part of the SysML's abstract syntax and are stated within the "Attributes" and "Constraints" parts of each subsection.

In order to support this rationale, the framework must consist of elements that are not ambiguous, are consistent with the language itself and are easily assessable with each candidate. For the framework to be successful, each element must be easily identified within a modelling tool's language implementation.

11.2 Omitted Elements

The following specification items were omitted from the evaluation framework and a reason for their omission is provided. In accordance with the rationale for producing the evaluation framework, sections 12, 13, 14 and 17 were not considered for incorporation.

11.2.1 Section 7: Model Elements

11.2.1.1 ViewElement (subsection 7.3.2.4) - Constraint 2

OMG states that "The precise semantic of this constraint is a semantic variation point."(OMG, 2006, p. 29). As this constraint does not precisely define a limitation on a View's structure, usage or implementation, this constraint was omitted from incorporation into the framework.

Its omission is in accordance with the previously stated rationale for the evaluation framework.

11.2.2 Section 9: Ports and Flows

11.2.2.1 FlowPort (subsection 9.3.2.5) - Constraint 1

Constraints 2 and 3 were retained instead of constraint 1 to ensure that the framework demands the existence of Atomic and Non-atomic FlowPorts and their type restrictions. Constraint 2 will require that a Non-atomic FlowPort must be typed by a FlowSpecification and constraint 3 will require that an Atomic FlowPort is typed by a Block, Signal, Datatype or ValueType.

11.2.2.2 FlowPort (subsection 9.3.2.5) - Constraint 2 in Section 9.3.2.6

The constraint in OMG states that "An *in* FlowProperty value cannot be modified by its owning Block." (OMG, 2006, p. 65). This constraint was not incorporated into the framework as this researcher was unable to determine how this constraint could be accurately verified within a modelling tool's language implementation.

Constraint 3 in Section 9.3.2.6

The constraint in OMG states that "An out FlowProperty cannot be read by its owning Block." (OMG, 2006, p. 65). This constraint was not incorporated into the framework as this researcher was unable to determine how this constraint could be accurately verified within a modelling tool's language implementation.

According to a discussion of this constraint by OMG (2007a), constraint 3 of section 9.3.2.6 has been identified by the SysML Finalisation Task Force (FTF) as a candidate for deletion and may be omitted from a future revision of the language.

11.2.3 Section 11: Activities

11.2.3.1 Overwrite (subsection 11.3.2.5) – Constraint 1

This constraint in OMG (2007a, p. 95) was omitted from the framework as it appears to repeat the contents of constraint 1 from section 11.3.2.4 for the "NoBuffer" stereotype.

11.2.4 Section 12 (entire section): Interactions

Due to a lack of SysML extensions within section 12 of the specification, this section has not been incorporated into the evaluation framework.

OMG (2006, p. 105) indicates the omission of the Communication, Interaction and Timing UML diagram types from the UML4SysML subset that is utilised by SysML. Section 12 of the specification indicates that no SysML extensions have been made for these diagram types. However, usage examples of Sequence diagrams have been provided.

11.2.5 Section 13 (entire section): State Machines

SysML and UML 2.1 defines generic state machines in the same way and protocol state machines have been omitted from the SysML language (OMG, 2006, p. 109). Section 13 of the specification states that no extensions have been considered for the SysML (OMG, 2006, p. 112).

This section was not considered appropriate for incorporation into the framework as it does not stipulate any SysML-specific extensions and instead relies on an existing UML implementation.

11.2.6 Section 14 (entire section): Use Cases

The OMG SysML 1.0a Final Adopted Specification (FAS) states that "There are no SysML extensions to UML 2.1 use cases." (OMG, 2006, p. 117). Since SysML merely reuses this UML diagram type, this section has been omitted from incorporation in accordance with the aforementioned rationale for producing the evaluation framework.

11.2.7 Section 16: Requirements

11.2.7.1 DeriveReqt (subsection 16.3.2.2) - Constraint 2

Since only elements stereotyped by «requirement» (or one its children) are permitted for client and supplier elements in a DeriveReqt relationship, the second constraint of the DeriveReqt stereotype, described in OMG (2007a, p. 144), was merged with its first constraint.

11.2.8 Section 17: Profiles & Model Libraries

This section was not considered for the evaluation framework as it relies entirely on an underlying implementation of the UML. OMG (2006) elaborates on how one may utilise the UML's profile mechanism and does not mention any modifications or extensions to its capabilities.

OMG (2006, p. 157) states that the SysML does not add any further elements to the profile or model library mechanism and no UML extensions have been stated.

12 Appendix B: Initial Framework

Table 12.1 shows the comparative framework before it was screened using the procedures in section 6.4.1.

Table 12.1:

The pilot comparative framework.

Section	Page	Diagram	Heading	Element	Category	Description
7.3.2.1	28	SysML Diagram	Conform	Conform	Rule	Conform extends from UML4SysML::Dependency metaclass.
7.3.2.1	28	SysML Diagram	Conform	Conform	Rule	Target element of a conform relationship must have a «viewpoint» stereotype.
7.3.2.1	28	SysML Diagram	Conform	Conform	Rule	Source element of a conform relationships must have a «view» stereotype.
7.3.2.2	28	SysML Diagram	Problem	Problem	Rule	Problem extends from UML4SysML::Comment metaclass.
7.3.2.3	29	SysML Diagram	Rationale	Rationale	Rule	Rationale extends from UML4SysML::Comment metaclass.
7.3.2.4	29	SysML Diagram	View	View	Attribute	viewpoint:Viewpoint[1] - contents are derived from the supplier of the view's «conform» dependency.
7.3.2.4	29	SysML Diagram	View	View	Rule	Views can only own the following element types: element import, package import, comment, and constraint
7.3.2.4	29	SysML Diagram	View	View	Rule	Constraint 2 was not included - subject to semantic variation.
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Attribute	stakeholders:String[*]Set of stakeholders.
7.3.2.5	29	SysML	Viewpoint	ViewPoint	Attribute	concerns:String[*] - the stakeholder's interests.

		Diagram				
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Attribute	purpose:String - addresses the stakeholder concerns.
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Attribute	languages:String[*] - the ViewPoint's languages
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Attribute	methods:String[*] - the methods used to construct the ViewPoint's views
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Rule	An instance specification cannot have a Viewpoint as its classifier.
7.3.2.5	29	SysML Diagram	Viewpoint	ViewPoint	Rule	The "ownedOperations" property must not contain a value.
7.3.2.5	30	SysML Diagram	Viewpoint	ViewPoint	Rule	The "ownedAttributes" property must not contain a value.
7.3.2.5	30	SysML Diagram	Viewpoint	ViewPoint	Rule	The "isAbstract" property must contain a "True" value.
8.3.1.2	40	Block Definition Diagram	Block and ValueType Definitions	Block	Rule	Block extends from UML4SysML::Class metaclass.
8.3.1.2	41	Block Definition Diagram	Default «block» stereotype on unlabeled box	Block	Rule	Unlabeled definition boxes have a default stereotype «block»
8.3.1.2	41	Block Definition Diagram	Namespace Compartment	Block	Rule	The Namespace compartment must show Blocks contained by current Block.
8.3.1.2	41	Block Definition Diagram	Structure Compartment	Block	Rule	The Structure compartment must show elements and connections internal to the Block definition.
8.3.1.2	42	Block Definition Diagram	Constraint Compartment	Block	Rule	Constraints may be shown as linked note with «constraint» stereotype.
8.3.1.2	42	Block Definition	Constraint Compartment	BlockProperty	Rule	Default multiplicity of 0..1 for any type of diamond end of part or shared associations.

		Diagram				
8.3.1.2	42	Block Definition Diagram	Default Multiplicities	BlockProperty	Rule	Default multiplicity of 1 for target end of unidirectional associations.
8.3.1.3	42	Internal Block Diagram	Property types	BlockProperty	Rule	Referenced properties are shown as a dash-outlined box.
8.3.1.3	43	Internal Block Diagram	Nested Connector End	Block	Rule	Boundary-crossing connectors may be established with a block's nested block properties.
8.3.1.3	43	Internal Block Diagram	Nested Connector End	Block	Rule	Unless additional properties are made available at each containing block, a connector end that is nested more than one level deep must have the stereotype «NestedConnectorEnd» automatically applied.
8.3.1.3	43	Internal Block Diagram	Property Path Name	Block	Rule	Multi-level property reference names for Internal Properties for Blocks.
8.3.1.3	43	Internal Block Diagram	Property-specific type	Block	Rule	A property with a local specialised type may show its type name in square brackets.
8.3.1.3	43	Internal Block Diagram	Property-specific type	Block	Rule	A property with no type signature in its definition has a property-specific type provided by local declarations.
8.3.1.3	43	Internal Block Diagram	Default value compartment	Block	Rule	"defaultValue" compartment for displaying the default value of a property
8.3.2.1	46	Internal Block Diagram	Constraints	Block	Rule	Exactly two(2) ends for associations with ends typed by blocks
8.3.2.1	46	Internal Block Diagram	Constraints	Block	Rule	Connector contains exactly two(2) ends

8.3.2.1	46	Internal Block Diagram	Constraints	Block	Rule	Omission of optional "name" property for UML Property metaclass instances (Class applying <<block>>) typed by block and owned by association
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	As part of UML profile: Property typed by block must be defined as an end of an association. An inverse of the connection must always be present.
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Attribute	isEncapsulated : Boolean [0..1]
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	If isEncapsulated is "True": only Port connections to parts typed by this Block are valid.
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	If isEncapsulated is "False": connections can be made to internal parts of parts typed by this Block using deep-nested connector ends.
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	[3]In the UML metamodel on which SysML is built, any instance of the Property metaclass that is typed by a block (a Class with the «block» stereotype applied) and which is owned by an Association may not have a name and may not be defined as a navigable owned end of the association. (While the Property has a "name" property as defined by its NamedElement superclass, the value of the "name" property, which is optional, must be missing.)
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	[4]In the UML metamodel on which SysML is built, a Property that is typed by a block must be defined as an end of an association. (An inverse end of this association, whether owned by another block or the association itself, must always be present so there is always a metamodel element to record the inverse multiplicity of the reference.)
8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	[5]The following constraint under Section 9.3.6, "Connector" in the UML 2.0 Superstructure Specification (OMG document formal/05-07-04) is removed by SysML: "[3] The ConnectableElements attached as roles to each ConnectorEnd owned by a Connector must be roles of the Classifier that owned the Connector, or they must be ports of such roles."

8.3.2.1	47	Internal Block Diagram	Constraints	Block	Rule	BlockProperty extends from UML4SysML::Property metaclass.
8.3.2.2	48	Internal Block Diagram	Constraints	BlockProperty	Rule	A property must be typed by a Block if it is an attribute with "composite" or "shared" aggregation.
8.3.2.3	48	Internal Block Diagram	DistributedProperty	DistributedProperty	Rule	DistributedProperty extends from SysML::BlockProperty stereotype.
8.3.2.4	48	Internal Block Diagram	Dimension	Dimension	Rule	Dimension stereotype extends from SysML::ValueType.
8.3.2.4	48	Internal Block Diagram	Constraints	Dimension	Rule	No values should be contained in the inherited ValueType attributes "dimension" and "unit".
8.3.2.5	48	Internal Block Diagram	NestedConnectorEnd	NestedConnectorEnd	Rule	NestedConnectorEnd extends from UML4SysML::ConnectorEnd metaclass.
8.3.2.5	48	Internal Block Diagram	NestedConnectorEnd	NestedConnectorEnd	Attribute	propertyPath: Property [2..*] (in order)
8.3.2.5	48	Internal Block Diagram	NestedConnectorEnd	NestedConnectorEnd	Rule	First property in propertyPath must be owned by block that owns the connector.
8.3.2.5	48	Structure Diagram	NestedConnectorEnd	NestedConnectorEnd	Rule	After the first property in the propertyPath list, every successive property must be contained in the block that types the property at the immediately preceding position.
8.3.2.6	48	SysML Diagram	Unit	Unit	Rule	The "unit" attribute of the ValueType stereotype cannot contain any value.
8.3.2.7	49	SysML Diagram	ValueType	ValueType	Rule	ValueType extends from UML4SysML::DataType.
8.3.2.7	49	SysML	ValueType	ValueType	Attribute	dimension: ValueType [0..1]

		Diagram				
8.3.2.7	49	SysML Diagram	ValueType	ValueType	Attribute	unit: ValueType [0..1]
8.3.2.7	49	SysML Diagram	ValueType	ValueType	Rule	"dimension" attribute must reference a ValueType stereotyped by «dimension».
8.3.2.7	49	SysML Diagram	ValueType	ValueType	Rule	"unit" attribute must reference a ValueType stereotyped by «unit».
8.3.2.7	49	SysML Diagram	ValueType	ValueType	Rule	If a value is defined for the "unit" attribute, the "dimension" attribute must be the same as the dimension property for the "unit" attribute.
8.3.3.1	50	SysML Diagram	Complex	Complex	Rule	Complex extends from SysML::Blocks::ValueType
8.3.3.1	50	SysML Diagram	Complex	Complex	Attribute	realPart: Real
8.3.3.1	50	SysML Diagram	Complex	Complex	Attribute	imaginaryPart: Real
8.3.3.2	50	SysML Diagram	Real	Real	Rule	Real extends from SysML::Blocks::ValueType
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Attribute	direction : FlowDirection [0..1]
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Attribute	isConjugated : Boolean [0..1]
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Attribute	isAtomic : Boolean (derived)
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Rule	A Block, Signal, DataType or ValueType can type an Atomic FlowPort.
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Rule	A FlowSpecification can type a Non-Atomic FlowPort.
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Rule	An Atomic FlowPort must have the following tagged value and property settings: isAtomic=True and the multiplicities of Direction=1 and isConjugated=0

9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Rule	A Non-Atomic FlowPort must have the following tagged value and property settings: isAtomic=False and the multiplicities of Direction=0 and isConjugated=1
9.3.2.5	64	SysML Diagram	FlowPort	FlowPort	Rule	Only FlowPorts of matching type, direction and name properties can share connections.
9.3.2.6	65	SysML Diagram	FlowProperty	FlowProperty	Attribute	direction : FlowDirection
9.3.2.6	65	SysML Diagram	FlowProperty	FlowProperty	Rule	FlowProperty elements can be typed by either a Block, Signal, ValueType or DataType.
9.3.2.7	65	SysML Diagram	FlowSpecification	FlowSpecification	Rule	FlowSpecifications cannot own operations or receptions, only FlowProperties.
9.3.2.8	66	SysML Diagram	ItemFlow	ItemFlow	Attribute	itemProperty: a BlockProperty [0..1].
9.3.2.8	66	SysML Diagram	ItemFlow	ItemFlow	Rule	ItemFlow can be assigned to connectors and associations.
9.3.2.8	66	SysML Diagram	ItemFlow	ItemFlow	Rule	A Block or a ValueType can type an ItemProperty.
9.3.2.8	66	SysML Diagram	ItemFlow	ItemProperty	Rule	An ItemProperty is defined in the context of the Block that owns its respective connector.
9.3.2.8	66	SysML Diagram	ItemFlow	ItemProperty	Rule	The type of an ItemProperty is the same as or a subclass of "conveyedClassifier".
9.3.2.8	66	SysML Diagram	ItemFlow	ItemProperty	Rule	For itemProperty, its muliplicity is 0 only if assigned to an association. It is only defined for ItemFlows assigned to connectors.
10.3.2.1	75	Structure Diagram	Stereotypes	ConstraintBlock	Rule	ConstraintBlocks can only own properties defining its constraint parameters, constraint properties holding internal usages of constraint blocks, binding connectors between its internally nested constraint parameters, constraint expressions that define an interpretation for the constraint block, and general-purpose model management and crosscutting elements.
10.3.2.2	75	Structure Diagram	Stereotypes	ConstaintProperty	Rule	A ConstraintProperty must be a BlockProperty of type ConstraintBlock.

11.3.1.1	90	Structure Diagram	Activity	Activity	Rule	For a composition association between activities: the name of a synchronous CallBehaviourAction (that is contained in an activity) and its part end must be the same. For Actions with no names that is only called (or used) once in an Activity, the name of the end is used for the name of the Activity.
11.3.1.1	90	Structure Diagram	CallBehaviourAction		Rule	For a composition association between activities: A CallBehaviourAction must contain (own) an Action with the same name as the Action or Activity that is a part end in a composition relationship.
11.3.1.1	90	Structure Diagram	Activity	Activity	Rule	For a composition association between activities: Part end must have a lower multiplicity of 0 (zero).
11.3.1.1	90	Structure Diagram	Activity	Activity	Rule	For a composition association between activities: If the activity engages nonreentrant behaviour, the Part end must have an upper multiplicity of 1 (one).
11.3.1.4	92	Structure Diagram	ObjectNode	ObjectNode	Rule	For associations defined between activities and classifiers typing object nodes: [1]The end name towards the object node type is the same as the name of an object node in the activity at the other end.
11.3.1.4	92	Structure Diagram	ObjectNode	ObjectNode	Rule	For associations defined between activities and classifiers typing object nodes: [2]The classifier must be the same as the type of the corresponding object node.
11.3.1.4	93	Structure Diagram	ObjectNode	ObjectNode	Rule	The object node type end must have a lower multiplicity of 0 (zero).
11.3.1.4	93	Structure Diagram	ObjectNode	ObjectNode	Rule	The object node type end must have an upper multiplicity equal to the upper bound of the corresponding object node.
11.3.2.2	94	Structure Diagram	ControlOperator	ControlOperator	Rule	Behaviour or operations with the «ControlOperator» stereotype applied must contain a parameter typed by ControlValue.
11.3.2.2	94	Structure Diagram	ControlOperator	ControlOperator	Rule	If a Behaviour is a method of an operation with the «ControlOperator» stereotype applied, it too must have the «ControlOperator» stereotype applied.
11.3.2.3	94	Structure Diagram	Discrete	Discrete	Rule	An element cannot have both «discrete» and «continuous» stereotypes applied.

11.3.2.4	94	Structure Diagram	NoBuffer	Nobuffer	Rule	An element cannot have both «overwrite» and «noBuffer» stereotypes applied.
11.3.2.5	95	Structure Diagram	Overwrite	Overwrite	Rule	An element cannot have both «noBuffer» and «overwrite» stereotypes applied.
11.3.2.6	95	Structure Diagram	Optional	Optional	Rule	If a parameter has the «optional» stereotype applied, its lower multiplicity value must be equal to 0 (zero), otherwise it must have a lower multiplicity value that is greater than zero (>0).
11.3.2.7	95	Structure Diagram	Probability	Probability	Rule	The «probability» stereotype is only applicable to ouput parameter sets or activity edges, which must have decision nodes or ObjectNodes as sources.
11.3.2.7	95	Structure Diagram	Probability	Probability	Rule	A «probability» stereotype applied to a single activity edge must be applied to all edges of a source activity.
11.3.2.7	95	Structure Diagram	Probability	Probability	Rule	A «probability» stereotype applied to an output parameter set must be applied to all output parameter sets for the behavour or operation owning the original output parameter set.
11.3.2.7	96	Structure Diagram	Probability	Probability	Rule	If the «probability» stereotype is applied to an output parameter set, all of its output parameters must belong to a parameter set.
11.3.2.8	96	Structure Diagram	Rate	Rate	Rule	[1]The value of the rate attribute must be an instance specification that is typed by a classifier that is stereotyped by SysML::«valueType» or SysML::«distributionDefinition».
11.3.2.8	96	Structure Diagram	Rate	Rate	Rule	[2]When the «rate» stereotype is applied to a parameter, the parameter must be streaming.
11.3.2.8	96	Structure Diagram	Rate	Rate	Rule	[3]The rate of a parameter must be less than or equal to rates on edges that come into or go out from pins and parameters nodes corresponding to the parameter.
11.3.2.10	96	Structure Diagram	ControlValue	ControlValue	Rule	ObjectNodes typed with "ControlValue" must have a "true" value for property UML4SysML::ObjectNode::isControlType
15.3.2.1	127	SysML Diagram	Allocate(from Allocations)	Allocate	Rule	Only one supplier(from) and one to many clients(to) allowed for an «allocate» dependancy. This also applies to its subclasses.
15.3.2.2	127	SysML Diagram	Allocated(from Allocations)	Allocated	Attribute	allocatedTo:NamedElement[*]: The union of all clients to which current instance is the supplier.

15.3.2.2	128	SysML Diagram	Allocated(from Allocations)	Allocated	Attribute	allocatedFrom:NamedElement[*]: the union of all suppers to which this is a client.
15.3.2.3	128	SysML Diagram	AllocateActivityPartition(from Allocations)	AllocateActivityPartition	Rule	With an AllocateActivityPartition, the supplier end of an «allocate» stereotyped dependency must be an Action and the client must be an AllocateActivityPartition.
16.3.2.1	143	Requirement Diagram	Copy(from Requirements)	Copy	Rule	Only two classes that are stereotyped with, or a relative subclass of, «requirement» can share a «copy» dependency.
16.3.2.1	143	Requirement Diagram	Copy(from Requirements)	Copy	Rule	All supplier requirements are copied to the client requirements. «copy» dependencies are made between all sub-requirements and the copy.
16.3.2.1	143	Requirement Diagram	Copy(from Requirements)	Copy	Rule	The supplier requirement's text property must be equal to the text property of the client requirement.
16.3.2.1	143	Requirement Diagram	Copy(from Requirements)	Copy	Rule	All sub-requirements recursively use the suppler's text property for the client's text property.
16.3.2.2	144	Requirement Diagram	DeriveReqt (from Requirements)	DeriveReqt	Rule	Clients and suppliers of a deriveReqt relationship must have the «requirement» stereotype or a subtype of «requirement» applied.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	text: String: The textual representation or a reference to the textual representation of the requirement.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	id: String: The unique id of the requirement.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/satisfiedBy: NamedElement[*]: Derived from all elements that are the client of a <<satisfy>> relationship for which this requirement is a supplier.

16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/verifiedBy: NamedElement[*]: Derived from all elements that are the client of a <<verify>> relationship for which this requirement is a supplier.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/tracedTo: NamedElement[*]: Derived from all elements that are the client of a <<trace>> relationship for which this requirement is a supplier.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/derived: Requirement[0..1]: Derived from all requirements that are the client of a <<deriveReqt>> relationship for which this requirement is a supplier.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/derivedFrom: Requirement[*]: Derived from all requirements that are the supplier of a <<deriveReqt>> relationship for which this requirement is a client.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/refinedBy: NamedElement[*]: Derived from all elements that are the client of a <<refine>> relationship for which this requirement is a supplier.
16.3.2.3	144	Requirement Diagram	Requirement (from Requirements)	Requirement	Attribute	/master: Requirement[0..1]: This is a derived property that lists the master requirement for this slave requirement. The master attribute is derived from the supplier of the Copy dependency that has this requirement as the slave.
16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	The "isAbstract" property must have a "true" value.
16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	The ownedOperation property must not have a value defined.
16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	The ownedAttribute property must not have a value defined.
16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	«requirement» stereotyped classes are unable to have association relationships.

16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	«requirement» stereotyped classes are unable to have generalisation relationships.
16.3.2.3	145	Requirement Diagram	Requirement (from Requirements)	Requirement	Rule	A «requirement» stereotyped class must have the «requirement» stereotype applied to all of its nested classifiers.
16.3.2.4	145	Requirement Diagram	RequirementRelated (from Requirements)	RequirementRelated	Attribute	\verifies: Requirement[*]: Derived from all requirements that are the supplier of a <<verify>> relationship for which this element is a client.
16.3.2.4	145	Requirement Diagram	RequirementRelated (from Requirements)	RequirementRelated	Attribute	\satisfies: Requirement[*]: Derived from all requirements that are the supplier of a <<satisfy>> relationship for which this element is a client.
16.3.2.4	145	Requirement Diagram	RequirementRelated (from Requirements)	RequirementRelated	Attribute	\refines: Requirement[*]: Derived from all requirements that are the supplier of a <<refine>> relationship for which this element is a client.
16.3.2.4	145	Requirement Diagram	RequirementRelated (from Requirements)	RequirementRelated	Attribute	\tracedFrom: Requirement[*]: Derived from all requirements that are the supplier of a <<trace>> relationship for which this element is a client.
16.3.2.5	145	Requirement Diagram	TestCase (from Requirements)	TestCase	Rule	The type of return parameter of the stereotyped model element must be VerdictKind. (note this is consistent with the UML Testing Profile).
16.3.2.6	146	Requirement Diagram	Satisfy (from Requirements)	Satisfy	Rule	Suppliers of a satisfy relationship must have the «requirement» stereotype or a subtype of «requirement» applied.
16.3.2.7	146	Requirement Diagram	Verify (from Requirements)	Verify	Rule	Suppliers of a verify relationship must have the «requirement» stereotype or a subtype of «requirement» applied.
16.3.2.7	146	Requirement Diagram	Verify (from Requirements)	Verify	Rule	Clients of a verify relationship must have the «testCase» stereotype or a subtype of «testCase» applied.

13 Appendix C: Comparative Framework Results

Table 13.1:

Framework and results for the comparative evaluation.

Section	Group	Element	Type	Description	EmbeddedPlus SysML Toolkit 2.0.0.2	MDG SysML Technology Add-In 6.5	Magicdraw SysML Plugin 1.1
7.3.2.1	Model Element	Conform	Rule	Conform extends from UML4SysML::Dependency metaclass.	TRUE	TRUE	TRUE
7.3.2.1	Model Element	Conform	Rule	Target element of a conform relationship must have a «viewpoint» stereotype.	TRUE	TRUE	TRUE
7.3.2.1	Model Element	Conform	Rule	Source element of a conform relationships must have a «view» stereotype.	TRUE	TRUE	TRUE
7.3.2.2	Model Element	Problem	Rule	Problem extends from UML4SysML::Comment metaclass.	TRUE	TRUE	TRUE
7.3.2.3	Model Element	Rationale	Rule	Rationale extends from UML4SysML::Comment metaclass.	TRUE	TRUE	TRUE
7.3.2.4	Model Element	View	Attribute	viewpoint:Viewpoint[1] - contents are derived from the supplier of the view's «conform» dependency.	TRUE	**FALSE**	PARTIAL

7.3.2.4	Model Element	View	Rule	Views can only own the following element types: element import, package import, comment, and constraint	**FALSE**	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Attribute	stakeholders:String[*]Set of stakeholders.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Attribute	concerns:String[*] - the stakeholder's interests.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Attribute	purpose:String - addresses the stakeholder concerns.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Attribute	languages:String[*] - the ViewPoint's languages	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Attribute	methods:String[*] - the methods used to construct the ViewPoint's views	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Rule	An instance specification cannot have a Viewpoint as its classifier.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Rule	The "ownedOperations" property must not contain a value.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Rule	The "ownedAttributes" property must not contain a value.	TRUE	TRUE	TRUE
7.3.2.5	Model Element	ViewPoint	Rule	The "isAbstract" property must contain a "True" value.	**FALSE**	TRUE	TRUE

8.3.1.2	Structure	Block	Rule	Block extends from UML4SysML::Class metaclass.	TRUE	TRUE	TRUE
8.3.1.2	Structure	Block	Rule	The Structure compartment must show elements and connections internal to the Block definition.	TRUE	TRUE	TRUE
8.3.1.3	Structure	Block	Rule	Unless additional properties are made available at each containing block, a connector end that is nested more than one level deep must have the stereotype «NestedConnectorEnd» automatically applied.	**FALSE**	**FALSE**	**FALSE**
8.3.2.1	Structure	Block	Rule	Exactly two(2) ends for associations with ends typed by blocks	TRUE	TRUE	TRUE
8.3.2.1	Structure	Block	Rule	Connector contains exactly two(2) ends	TRUE	TRUE	TRUE
8.3.2.1	Structure	Block	Attribute	isEncapsulated : Boolean [0..1]	TRUE	TRUE	TRUE
8.3.2.1	Structure	Block	Rule	If isEncapsulated is "True": only Port connections to parts typed by this Block are valid.	**FALSE**	**FALSE**	**FALSE**
8.3.2.1	Structure	Block	Rule	If isEncapsulated is "False": connections can be made to internal parts of parts typed by this Block using deep-nested connector ends.	**FALSE**	TRUE	**FALSE**
8.3.2.1	Structure	Block	Rule	[3]In the UML metamodel on which SysML is built, any instance of the Property metaclass that is typed by a block (a Class with the «block» stereotype applied) and which is owned by an Association may not have a name and may not be defined as a navigable owned end of the association. (While the Property has a "name"	TRUE	**FALSE**	TRUE

				property as defined by its NamedElement superclass, the value of the "name" property, which is optional, must be missing.)			
8.3.2.1	Structure	Block	Rule	[4]In the UML metamodel on which SysML is built, a Property that is typed by a block must be defined as an end of an association. (An inverse end of this association, whether owned by another block or the association itself, must always be present so there is always a metamodel element to record the inverse multiplicity of the reference.)	TRUE	TRUE	TRUE
8.3.2.1	Structure	Block	Rule	[5]The following constraint under Section 9.3.6, "Connector" in the UML 2.0 Superstructure Specification (OMG document formal/05-07-04) is removed by SysML: "[3] The ConnectableElements attached as roles to each ConnectorEnd owned by a Connector must be roles of the Classifier that owned the Connector, or they must be ports of such roles."	TRUE	TRUE	TRUE
8.3.2.1	Structure	Block	Rule	BlockProperty extends from UML4SysML::Property metaclass.	TRUE	TRUE	TRUE
8.3.2.2	Structure	BlockProperty	Rule	A property must be typed by a Block if it is an attribute with "composite" or "shared" aggregation.	**FALSE**	**FALSE**	**FALSE**
8.3.2.3	Structure	DistributedProperty	Rule	DistributedProperty extends from SysML::BlockProperty stereotype.	TRUE	**FALSE**	TRUE

8.3.2.4	Structure	Dimension	Rule	Dimension stereotype extends from SysML::ValueType.	**FALSE**	TRUE	TRUE
8.3.2.4	Structure	Dimension	Rule	No values should be contained in the inherited ValueType attributes "dimension" and "unit".	**FALSE**	TRUE	TRUE
8.3.2.6	Structure	Unit	Rule	The "unit" attribute of the ValueType stereotype cannot contain any value.	**FALSE**	TRUE	TRUE
8.3.2.7	Structure	ValueType	Rule	ValueType extends from UML4SysML::DataType.	TRUE	TRUE	TRUE
8.3.2.7	Structure	ValueType	Attribute	dimension: ValueType [0..1]	TRUE	TRUE	TRUE
8.3.2.7	Structure	ValueType	Attribute	unit: ValueType [0..1]	TRUE	TRUE	TRUE
8.3.2.7	Structure	ValueType	Rule	"dimension" attribute must reference a ValueType stereotyped by «dimension».	PARTIAL	TRUE	TRUE
8.3.2.7	Structure	ValueType	Rule	"unit" attribute must reference a ValueType stereotyped by «unit».	PARTIAL	TRUE	TRUE
8.3.2.7	Structure	ValueType	Rule	If a value is defined for the "unit" attribute, the "dimension" attribute must be the same as the dimension property for the "unit" attribute.	TRUE	TRUE	TRUE
8.3.3.1	Structure	Complex	Rule	Complex extends from SysML::Blocks::ValueType	**FALSE**	**FALSE**	TRUE
8.3.3.1	Structure	Complex	Attribute	realPart: Real	TRUE	**FALSE**	TRUE
8.3.3.1	Structure	Complex	Attribute	imaginaryPart: Real	TRUE	**FALSE**	TRUE

8.3.3.2	Structure	Real	Rule	Real extends from SysML::Blocks::ValueType	**FALSE**	**FALSE**	TRUE
9.3.2.5	Structure	FlowPort	Attribute	direction : FlowDirection [0..1]	TRUE	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Attribute	isConjugated : Boolean [0..1]	TRUE	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Attribute	isAtomic : Boolean (derived)	TRUE	**FALSE**	TRUE
9.3.2.5	Structure	FlowPort	Rule	A Block, Signal, DataType or ValueType can type an Atomic FlowPort.	TRUE	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Rule	A FlowSpecification can type a Non-Atomic FlowPort.	TRUE	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Rule	An Atomic FlowPort must have the following tagged value and property settings: isAtomic=True and the multiplicities of Direction=1 and isConjugated=0	**FALSE**	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Rule	A Non-Atomic FlowPort must have the following tagged value and property settings: isAtomic=False and the multiplicities of Direction=0 and isConjugated=1	TRUE	TRUE	TRUE
9.3.2.5	Structure	FlowPort	Rule	Only FlowPorts of matching type, direction and name properties can share connections.	**FALSE**	TRUE	**FALSE**
9.3.2.6	Structure	FlowProperty	Attribute	direction : FlowDirection	TRUE	TRUE	TRUE
9.3.2.6	Structure	FlowProperty	Rule	FlowProperty elements can be typed by either a Block, Signal, ValueType or DataType.	TRUE	TRUE	TRUE

9.3.2.7	Structure	FlowSpecification	Rule	FlowSpecifications cannot own operations or receptions, only FlowProperties.	TRUE	TRUE	PARTIAL
9.3.2.8	Structure	ItemFlow	Attribute	itemProperty: a BlockProperty [0..1].	**FALSE**	TRUE	TRUE
9.3.2.8	Structure	ItemFlow	Rule	ItemFlow can be assigned to connectors and associations.	**FALSE**	TRUE	TRUE
9.3.2.8	Structure	ItemFlow	Rule	A Block or a ValueType can type an ItemProperty.	**FALSE**	**FALSE**	TRUE
9.3.2.8	Structure	ItemProperty	Rule	An ItemProperty is defined in the context of the Block that owns its respective connector.	**FALSE**	TRUE	**FALSE**
9.3.2.8	Structure	ItemProperty	Rule	The type of an ItemProperty is the same as or a subclass of "conveyedClassifier".	**FALSE**	**FALSE**	**FALSE**
9.3.2.8	Structure	ItemProperty	Rule	For itemProperty, its muliplicity is 0 only if assigned to an association. It is only defined for ItemFlows assigned to connectors.	**FALSE**	TRUE	TRUE
10.3.2.2	Parametrics	ConstaintProperty	Rule	A ConstraintProperty must be a BlockProperty of type ConstraintBlock.	**FALSE**	TRUE	TRUE
11.3.1.1	Behaviour	Activity	Rule	For a composition association between activities: the name of a synchronous CallBehaviourAction (that is contained in an activity) and its part end must be the same. For Actions with no names that is only called (or used) once in an Activity, the name of the end is used for the name of the Activity.	**FALSE**	**FALSE**	**FALSE**

11.3.1.1	Behaviour	Activity	Rule	For a composition association between activities: A CallBehaviourAction must contain (own) an Action with the same name as the Action or Activity that is a part end in a composition relationship.	TRUE	TRUE	**FALSE**
11.3.1.1	Behaviour	Activity	Rule	For a composition association between activities: Part end must have a lower multiplicity of 0 (zero).	TRUE	**FALSE**	**FALSE**
11.3.1.1	Behaviour	Activity	Rule	For a composition association between activities: If the activity engages nonreentrant behaviour, the Part end must have an upper multiplicity of 1 (one).	**FALSE**	**FALSE**	**FALSE**
11.3.1.4	Behaviour	ObjectNode	Rule	For associations defined between activities and classifiers typing object nodes: [1]The end name towards the object node type is the same as the name of an object node in the activity at the other end.	**FALSE**	**FALSE**	**FALSE**
11.3.1.4	Behaviour	ObjectNode	Rule	For associations defined between activities and classifiers typing object nodes: [2]The classifier must be the same as the type of the corresponding object node.	**FALSE**	**FALSE**	**FALSE**
11.3.1.4	Behaviour	ObjectNode	Rule	The object node type end must have a lower multiplicity of 0 (zero).	**FALSE**	**FALSE**	**FALSE**
11.3.1.4	Behaviour	ObjectNode	Rule	The object node type end must have an upper multiplicity equal to the upper bound of the corresponding object node.	**FALSE**	**FALSE**	**FALSE**

11.3.2.2	Behaviour	ControlOperator	Rule	Behaviour or operations with the «ControlOperator» stereotype applied must contain a parameter typed by ControlValue.	**FALSE**	TRUE	TRUE
11.3.2.2	Behaviour	ControlOperator	Rule	If a Behaviour is a method of an operation with the «ControlOperator» stereotype applied, it too must have the «ControlOperator» stereotype applied.	TRUE	TRUE	TRUE
11.3.2.3	Behaviour	Discrete	Rule	An element cannot have both «discrete» and «continuous» stereotypes applied.	TRUE	**FALSE**	TRUE
11.3.2.4	Behaviour	Nobuffer	Rule	An element cannot have both «overwrite» and «noBuffer» stereotypes applied.	TRUE	**FALSE**	TRUE
11.3.2.6	Behaviour	Optional	Rule	If a parameter has the «optional» stereotype applied, its lower multiplicity value must be equal to 0 (zero), otherwise it must have a lower multiplicity value that is greater than zero (>0).	**FALSE**	TRUE	TRUE
11.3.2.7	Behaviour	Probability	Rule	The «probability» stereotype is only applicable to ouput parameter sets or activity edges, which must have decision nodes or ObjectNodes as sources.	**FALSE**	TRUE	TRUE
11.3.2.7	Behaviour	Probability	Rule	A «probability» stereotype applied to a single activity edge must be applied to all edges of a source activity.	**FALSE**	**FALSE**	TRUE
11.3.2.7	Behaviour	Probability	Rule	A «probability» stereotype applied to an output parameter set must be applied to all output parameter sets for the behavour or operation	**FALSE**	**FALSE**	**FALSE**

				owning the original output parameter set.			
11.3.2.7	Behaviour	Probability	Rule	If the «probability» stereotype is applied to an output parameter set, all of its output parameters must belong to a parameter set.	**FALSE**	**FALSE**	**FALSE**
11.3.2.8	Behaviour	Rate	Rule	[1]The value of the rate attribute must be an instance specification that is typed by a classifier that is stereotyped by SysML::«valueType» or SysML::«distributionDefinition».	TRUE	**FALSE**	**FALSE**
11.3.2.8	Behaviour	Rate	Rule	[2]When the «rate» stereotype is applied to a parameter, the parameter must be streaming.	**FALSE**	**FALSE**	TRUE
11.3.2.8	Behaviour	Rate	Rule	[3]The rate of a parameter must be less than or equal to rates on edges that come into or go out from pins and parameters nodes corresponding to the parameter.	**FALSE**	**FALSE**	**FALSE**
11.3.2.10	Behaviour	ControlValue	Rule	ObjectNodes typed with "ControlValue" must have a "true" value for property UML4SysML::ObjectNode::isControlType	**FALSE**	**FALSE**	TRUE
15.3.2.1	Allocation	Allocate	Rule	Only one supplier(from) and one to many clients(to) allowed for an «allocate» dependancy. This also applies to its subclasses.	TRUE	TRUE	TRUE
15.3.2.2	Allocation	Allocated	Attribute	allocatedTo:NamedElement[*]: The union of all clients to which current instance is the supplier.	PARTIAL	PARTIAL	TRUE
15.3.2.2	Allocation	Allocated	Attribute	allocatedFrom:NamedElement[*]: the union of all suppers to which this is a client.	PARTIAL	PARTIAL	TRUE

15.3.2.3	Allocation	AllocateActivityPartition	Rule	With an AllocateActivityPartition, the supplier end of an «allocate» stereotyped dependency must be an Action and the client must be an AllocateActivityPartition.	**FALSE**	**FALSE**	**FALSE**
16.3.2.1	Requirements	Copy	Rule	Only two classes that are stereotyped with, or a relative subclass of, «requirement» can share a «copy» dependency.	TRUE	TRUE	TRUE
16.3.2.1	Requirements	Copy	Rule	All supplier requirements are copied to the client requirements. «copy» dependencies are made between all sub-requirements and the copy.	TRUE	**FALSE**	**FALSE**
16.3.2.1	Requirements	Copy	Rule	The supplier requirement's text property must be equal to the text property of the client requirement.	TRUE	**FALSE**	TRUE
16.3.2.1	Requirements	Copy	Rule	All sub-requirements recursively use the suppler's text property for the client's text property.	TRUE	**FALSE**	**FALSE**
16.3.2.2	Requirements	DeriveReqt	Rule	Clients and suppliers of a deriveReqt relationship must have the «requirement» stereotype or a subtype of «requirement» applied.	TRUE	**FALSE**	TRUE
16.3.2.3	Requirements	Requirement	Attribute	text: String: The textual representation or a reference to the textual representation of the requirement.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Attribute	id: String: The unique id of the requirement.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Attribute	/satisfiedBy: NamedElement[*]: Derived from all elements that are the client of a <<satisfy>>	TRUE	**FALSE**	TRUE

				relationship for which this requirement is a supplier.			
16.3.2.3	Requirements	Requirement	Attribute	/verifiedBy: NamedElement[*]: Derived from all elements that are the client of a <<verify>> relationship for which this requirement is a supplier.	TRUE	**FALSE**	PARTIAL
16.3.2.3	Requirements	Requirement	Attribute	/tracedTo: NamedElement[*]: Derived from all elements that are the client of a <<trace>> relationship for which this requirement is a supplier.	TRUE	**FALSE**	TRUE
16.3.2.3	Requirements	Requirement	Attribute	/derived: Requirement[0..1]: Derived from all requirements that are the client of a <<deriveReqt>> relationship for which this requirement is a supplier.	TRUE	**FALSE**	PARTIAL
16.3.2.3	Requirements	Requirement	Attribute	/derivedFrom: Requirement[*]: Derived from all requirements that are the supplier of a <<deriveReqt>> relationship for which this requirement is a client.	TRUE	**FALSE**	TRUE
16.3.2.3	Requirements	Requirement	Attribute	/refinedBy: NamedElement[*]: Derived from all elements that are the client of a <<refine>> relationship for which this requirement is a supplier.	TRUE	**FALSE**	TRUE
16.3.2.3	Requirements	Requirement	Attribute	/master: Requirement[0..1]: This is a derived property that lists the master requirement for this slave requirement. The master attribute is derived from the supplier of the Copy dependency that has this requirement as the	TRUE	**FALSE**	TRUE

				slave.			
16.3.2.3	Requirements	Requirement	Rule	The "isAbstract" property must have a "true" value.	**FALSE**	**FALSE**	TRUE
16.3.2.3	Requirements	Requirement	Rule	The ownedOperation property must not have a value defined.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Rule	The ownedAttribute property must not have a value defined.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Rule	«requirement» stereotyped classes are unable to have association relationships.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Rule	«requirement» stereotyped classes are unable to have generalisation relationships.	TRUE	TRUE	TRUE
16.3.2.3	Requirements	Requirement	Rule	A «requirement» stereotyped class must have the «requirement» stereotype applied to all of its nested classifiers.	**FALSE**	**FALSE**	TRUE
16.3.2.4	Requirements	RequirementRelated	Attribute	\verifies: Requirement[*]: Derived from all requirements that are the supplier of a <<verify>> relationship for which this element is a client.	TRUE	**FALSE**	TRUE
16.3.2.4	Requirements	RequirementRelated	Attribute	\satisfies: Requirement[*]: Derived from all requirements that are the supplier of a <<satisfy>> relationship for which this element is a client.	TRUE	**FALSE**	TRUE
16.3.2.4	Requirements	RequirementRelated	Attribute	\refines: Requirement[*]: Derived from all	TRUE	**FALSE**	TRUE

				requirements that are the supplier of a <<refine>> relationship for which this element is a client.			
16.3.2.4	Requirements	RequirementRelated	Attribute	\tracedFrom: Requirement[*]: Derived from all requirements that are the supplier of a <<trace>> relationship for which this element is a client.	TRUE	**FALSE**	TRUE
16.3.2.5	Requirements	TestCase	Rule	The type of return parameter of the stereotyped model element must be VerdictKind. (note this is consistent with the UML Testing Profile).	**FALSE**	**FALSE**	TRUE
16.3.2.6	Requirements	Satisfy	Rule	Suppliers of a satisfy relationship must have the «requirement» stereotype or a subtype of «requirement» applied.	TRUE	TRUE	TRUE
16.3.2.7	Requirements	Verify	Rule	Suppliers of a verify relationship must have the «requirement» stereotype or a subtype of «requirement» applied.	TRUE	TRUE	TRUE
16.3.2.7	Requirements	Verify	Rule	Clients of a verify relationship must have the «testCase» stereotype or a subtype of «testCase» applied.	TRUE	TRUE	TRUE

14 Appendix D: Qualitative Evaluation Results

14.1 *EmbeddedPlus Tool*

The following table contains the validation results obtained from the EmbeddedPlus tool during the qualitative evaluation.

Table 14.1:

Validation results from the EmbeddedPlus tool for the modelling problem.

Severity and Description	**Location**
IRJA0045E "<Activity Parameter Node> drivePower" must have an "out", "inout", or "return" parameter because it has incoming edges.	HybridSUV Model::HSUVModel::HSUVBehavior::Accelerate::drivePower
Assembly connector '<<bindingConnector>> <Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<<bindingConnector>> <Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<<bindingConnector>> <Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<<bindingConnector>> <Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<Connector> acl-ecu' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::acl-ecu

Assembly connector '<Connector> b-c' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: HybridSUV::b-c
Assembly connector '<Connector> b-i' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: HybridSUV::b-i
Assembly connector '<Connector> b-l' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::b-l
Assembly connector '<Connector> bk-ecu' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::bk-ecu
Assembly connector '<Connector> bk-l' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::bk-l
Assembly connector '<Connector> c-bk' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::c-bk
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::AccelerationEquation::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::PositionEquation::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::PowerEquation::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::VelocityEquation::Connector1
Assembly connector '<Connector> Connector1' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::InternalCombustionEngine::Connector1
Assembly connector '<Connector> Connector1' must only be defined	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::

from a role end requiring an interface to a role end providing that interface.	PowerParts::PowerSubSystem::Connector1
Assembly connector '<Connector> Connector10' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector10
Assembly connector '<Connector> Connector11' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector11
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::AccelerationEquation::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::PositionEquation::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::PowerEquation::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::VelocityEquation::Connector2
Assembly connector '<Connector> Connector2' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::InternalCombustionEngine::Connector2
Assembly connector '<Connector> Connector3' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::AccelerationEquation::Connector3
Assembly connector '<Connector> Connector3' must only be defined from a role end requiring an interface to a role end providing that	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector3

interface.	
Assembly connector '<Connector> Connector3' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector3
Assembly connector '<Connector> Connector3' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::InternalCombustionEngine::Connector3
Assembly connector '<Connector> Connector4' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector4
Assembly connector '<Connector> Connector4' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector4
Assembly connector '<Connector> Connector4' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::InternalCombustionEngine::Connector4
Assembly connector '<Connector> Connector5' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector5
Assembly connector '<Connector> Connector5' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector5
Assembly connector '<Connector> Connector5' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::InternalCombustionEngine::Connector5
Assembly connector '<Connector> Connector5' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::Connector5
Assembly connector '<Connector> Connector6' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector6
Assembly connector '<Connector> Connector6' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector6

Assembly connector '<Connector> Connector6' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::InternalCombustionEngine::Connector6
Assembly connector '<Connector> Connector6' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::Connector6
Assembly connector '<Connector> Connector7' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector7
Assembly connector '<Connector> Connector7' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics::Connector7
Assembly connector '<Connector> Connector7' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::Connector7
Assembly connector '<Connector> Connector8' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector8
Assembly connector '<Connector> Connector8' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::Connector8
Assembly connector '<Connector> Connector9' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVAnalysis::MOE::Connector9
Assembly connector '<Connector> epc-can' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::epc-can
Assembly connector '<Connector> epc' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::epc
Assembly connector '<Connector> fuelline' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::fuelline
Assembly connector '<Connector> i-l' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::i-l
Assembly connector '<Connector> ice' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::ice

Assembly connector '<Connector> p-bk' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::p-bk
Assembly connector '<Connector> p-c' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::HybridSUV::p-c
Assembly connector '<Connector> pumpToFuelOut' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::FuelTankAssembly::pumpToFuelOut
Assembly connector '<Connector> trsm-can' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::trsm-can
Assembly connector '<Connector> trsm' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::trsm
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::Diagrams::Collaboration1::Start Vehicle Black Box::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::AutomotiveDomain::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::AutomotiveDomain::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::AutomotiveDomain::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::AutomotiveDomain::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::AutomotiveDomain::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>
Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts:: PowerParts::PowerSubSystem::<Connector>

Assembly connector '<Connector>' must only be defined from a role end requiring an interface to a role end providing that interface.	HybridSUV Model::HSUVModel::HSUVStructure::VehicleParts::PowerParts::PowerSubSystem::<Connector>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> InternalCombustionEngine" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "client" of "<<allocate>> <Abstraction> PowerControlUnit" has 0 value(s); requires between 1 and (unlimited).	HybridSUV Model::HSUVModel::HSUVBehavior::<Abstraction>
IRJA0131W Property "event" of "<Message Occurrence Specification>" has 0 value(s); requires between 1 and 1.	HybridSUV Model::Diagrams::Collaboration1::Start Vehicle Black Box::<MessageOccurrenceSpecification>
IRJA0131W Property "refersTo" of "<Interaction Use> accelerate/cruise" has 0 value(s); requires between 1 and 1.	HybridSUV Model::Diagrams::Collaboration1::Interaction1::<CombinedFragment>::<InteractionOperand>::controlSpeed::<InteractionOperand>::accelerate/cruise
IRJA0131W Property "refersTo" of "<Interaction Use> brake" has 0 value(s); requires between 1 and 1.	HybridSUV Model::Diagrams::Collaboration1::Interaction1::<CombinedFragment>::<InteractionOperand>::controlSpeed::<InteractionOperand>::brake
IRJA0131W Property "refersTo" of "<Interaction Use> steer" has 0 value(s); requires between 1 and 1.	HybridSUV Model::Diagrams::Collaboration1::Interaction1::<CombinedFragment>::<InteractionOperand>::steer
IRJA0131W Property "signal" of "<Send Signal Event>" has 0 value(s); requires between 1 and 1.	HybridSUV Model::Diagrams::<SendSignalEvent>
IRJA0131W Property "value" of "<Value Pin>" has 0 value(s); requires between 1 and 1.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::ProportionPower::<ValuePin>

IRJA0137W "<<constraintBlock>> <Class> StraightLineVehicleDynamics" contains two or more indistinguishable members named "<<constraintProperty, acc>> <Property> acc."	HybridSUV Model::HSUVModel::HSUVAnalysis::StraightLineVehicleDynamics
IRJA0045E "<Activity Parameter Node> transModeCmd" must have an "out", "inout", or "return" parameter because it has incoming edges.	HybridSUV Model::HSUVModel::HSUVBehavior::Accelerate::transModeCmd
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
IRJA0064E "<<continuous>> <Object Flow>" must not have an action at either end.	HybridSUV Model::HSUVModel::HSUVBehavior::ProvidePower::<ObjectFlow>
Multiplicity element '<Property> testcase max acceleration : TestCase Max Acceleration [0..0]' does not define at least one valid cardinality that is greater than zero.	HybridSUV Model::HSUVModel::TestCases::<Association>::testcase max acceleration
The interface '<<flowSpecification>> <Interface> FS_ICE' has at least one feature which is not public.	HybridSUV Model::HSUVModel::HSUVStructure::HSUVInterfaces::FS_ICE
The interface '<<flowSpecification>> <Interface> TorqueFlow' has at least one feature which is not public.	HybridSUV Model::HSUVModel::Interfaces::TorqueFlow

14.2 Sparx Systems Tool

The following table contains the validation results obtained from the Sparx Systems tool during the qualitative evaluation.

Table 14.2:

Validation results from the Sparx Systems tool for the modelling problem.

MVR800005 - warning (epc :ElectricalPowerController (FlowPort)): Connected FlowPorts must have matching or compatible types
MVR800005 - warning (fuelFitting :Fuel (FlowPort)): Connected FlowPorts must have matching or compatible types
MVR800005 - warning (Port :FuelTankFitting (FlowPort)): Connected FlowPorts must have matching or compatible types
MVR800007 - error (PowerSubsystem::I_ICEData (FlowSpecification)): FlowSpecifications can only own FlowProperties
MVR800007 - error (PowerSubsystem::I_ICECmds (FlowSpecification)): FlowSpecifications can only own FlowProperties
MVR7F0001 - warning (Braking (Requirement)): Braking is unrealized
MVR7F0001 - warning (FuelEconomy (Requirement)): FuelEconomy is unrealized
MVR7F0001 - warning (OffRoadCapability (Requirement)): OffRoadCapability is unrealized
MVR7F0001 - warning (Acceleration (Requirement)): Acceleration is unrealized
MVR80001a - error (Automotive Value Types::Real (ValueType)): The dimension tagged value must reference a dimension
MVR80001a - error (Automotive Value Types::Real (ValueType)): The unit tagged value must reference a unit
MVR80001a - error (Automotive Value Types::Torque (ValueType)): The dimension tagged value must reference a dimension
MVR80001a - error (Automotive Value Types::Torque (ValueType)): The unit tagged value must reference a unit
MVR7F0001 - warning (PowerSourceManagement (Requirement)): PowerSourceManagement is unrealized
MVR7F0001 - warning (PowerSourceManagement (Requirement)): PowerSourceManagement is unrealized
MVR7F0001 - warning (Range (Requirement)): Range is unrealized
MVR7F0001 - warning (RegenerativeBraking (Requirement)): RegenerativeBraking is unrealized
MVR7F0001 - warning (Fuel Capacity (Requirement)): Fuel Capacity is unrealized
MVR7F0001 - warning (Cargo Capacity (Requirement)): Cargo Capacity is unrealized
MVR7F0001 - warning (Capacity (Requirement)): Capacity is unrealized

MVR7F0001 - warning (Passenger Capacity (Requirement)): Passenger Capacity is unrealized
MVR7F0001 - warning (Eco-Friendliness (Requirement)): Eco-Friendliness is unrealized
MVR7F0001 - warning (Ergonomics (Requirement)): Ergonomics is unrealized
MVR7F0001 - warning (Qualification (Requirement)): Qualification is unrealized
MVR7F0001 - warning (Safety Test (Requirement)): Safety Test is unrealized
MVR800006 - error (engineData :I_ICEData (FlowProperty)): A FlowProperty must be typed by a ValueType, DataType, Block or Signal
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR80000a - warning (<anonymous> (Association)): For composite association between behaviors, the part end name must match an action contained at the whole end
MVR050002 - error (<anonymous> (ControlFlow)): ControlFlow is not legal for StateNode --> InterruptibleActivityRegion
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for driveCurrent --> ProvideElectricPower
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ProvideElectricPower --> elecDrivePower
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for eThrottle --> ControlElectricPower
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ControlElectricPower --> driveCurrent
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for gThrottle --> ProvideGasPower
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ProvideGasPower --> gasDrivePower
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ProportionPower --> gThrottle
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ProportionPower --> eThrottle
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for battCond --> ActionPin
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for accelPosition --> ActionPin
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for ActionPin --> transModeCmd
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for speed --> ActionPin

MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for vehCond --> Decision
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for Decision --> speed
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for Decision --> battCond
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for elecDrivePower --> Decision
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for gasDrivePower --> Decision
MVR050002 - error (<anonymous> (ObjectFlow)): ObjectFlow is not legal for Decision --> drivePower
MVR7F0001 - warning (PowerSource Management (Requirement)): PowerSource Management is unrealized
Validation complete - 26 error(s), 26 warning(s)

14.3 Magicdraw Tool

The following table contains the validation results obtained from the Magicdraw tool during the qualitative evaluation.

Table 14.3:

Validation results from the Magicdraw tool for the modelling problem.

(not general.oclIsKindOf(SysML_Profile::Requirement)) and (not specific.oclIsKindOf(SysML_Profile::Requirement)) [MD Customization for SysML::SysML constraints::Requirement::Requirement1]	error	NXO	Non-executable:Semantic error::Classifier name expected (found implementation type tudresden.ocl20.core.parser.astgen.NamedElement, which is not a classifier implementation)
Range [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
-FuelPressure : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::Fuel]	warning	TpBlc	The type of the block property must be a block.
-incline : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::Road]	warning	TpBlc	The type of the block property must be a block.

CargoCapacity [HSUV Model::HSUVRequirements::HSUVSpecification::Capacity]	warning	RqAbst	The property isAbstract must be set to true.
ms^2 [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
F [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
PassengerCapacity [HSUV Model::HSUVRequirements::HSUVSpecification::Capacity]	warning	RqAbst	The property isAbstract must be set to true.
Pavement friction [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
mph [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
Power [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
Vehicle conditions [HSUV Model::HSUVRequirements::Adhesion utilization]	warning	RqAbst	The property isAbstract must be set to true.
FuelCapacity [HSUV Model::HSUVRequirements::HSUVSpecification::Capacity]	warning	RqAbst	The property isAbstract must be set to true.
OffRoadCapability [HSUV Model::HSUVRequirements::HSUVSpecification::Performance]	warning	RqAbst	The property isAbstract must be set to true.
mm [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
Eco-Friendliness [HSUV Model::HSUVRequirements::HSUVSpecification]	warning	RqAbst	The property isAbstract must be set to true.
-Position : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::HybridSUV]	warning	TpBlc	The type of the block property must be a block.
hp [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
SafetyTest [HSUV Model::HSUVRequirements::HSUVSpecification::Qualification]	warning	RqAbst	The property isAbstract must be set to true.
Braking [HSUV Model::HSUVRequirements::HSUVSpecification::Performance]	warning	RqAbst	The property isAbstract must be set to true.
Performance [HSUV Model::HSUVRequirements::HSUVSpecification]	warning	RqAbst	The property isAbstract must be set to true.
+fuelDemand : SysML Profile::Blocks::Real [HSUV	warning	TpBlc	The type of the block property must be a block.

Model::HSUVStructure::FuelInjector]			
-PayloadCapacity : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::HybridSUV]	warning	TpBlc	The type of the block property must be a block.
PerformanceView [HSUV Model::HSUVViews]	warning	VwOwn	A view can only own element import, package import, comment, and constraint elements.
-motorEfficiency : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::ElectricalMotorGenerator]	warning	TpBlc	The type of the block property must be a block.
-mpg : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::HybridSUV]	warning	TpBlc	The type of the block property must be a block.
PowerSourceManagement [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
RegenerativeBraking [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
sec [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
+fuelFlowRate : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::FuelTankAssembly]	warning	TpBlc	The type of the block property must be a block.
Adhesion utilization [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
ASTM R1337-90 [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
-VehicleDryWeight : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::HybridSUV]	warning	TpBlc	The type of the block property must be a block.
psi [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
Acceleration [HSUV Model::HSUVRequirements::HSUVSpecification::Performance]	warning	RqAbst	The property isAbstract must be set to true.
ft^3 [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
-generatorEfficiency : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::ElectricalMotorGenerator]	warning	TpBlc	The type of the block property must be a block.
Enable on Brake [HSUV Model::HSUVBehavior::Operate Car]	warning	CtrlOpPrm	When the «controlOperator» stereotype is applied, the behavior or operation must have at least one parameter typed by ControlValue.
Emissions [HSUV Model::HSUVRequirements::HSUVSpecification::Eco-Friendliness]	warning	RqAbst	The property isAbstract must be set to true.
-FuelWeight : SysML Profile::Blocks::Real [HSUV	warning	TpBlc	The type of the block property must be a block.

Model::HSUVStructure::FuelTankAssembly]			
Master Cylinder Efficacy [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
Qualification [HSUV Model::HSUVRequirements::HSUVSpecification]	warning	RqAbst	The property isAbstract must be set to true.
Ergonomics [HSUV Model::HSUVRequirements::HSUVSpecification]	warning	RqAbst	The property isAbstract must be set to true.
-iceEfficiency : SysML Profile::Blocks::Real [HSUV Model::HSUVStructure::InternalCombustionEngine]	warning	TpBlc	The type of the block property must be a block.
ft [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
FuelEconomy [HSUV Model::HSUVRequirements::HSUVSpecification::Performance]	warning	RqAbst	The property isAbstract must be set to true.
Reservoir [HSUV Model::HSUVRequirements]	warning	RqAbst	The property isAbstract must be set to true.
Test and procedure conditions [HSUV Model::HSUVRequirements::Adhesion utilization]	warning	RqAbst	The property isAbstract must be set to true.
lb [HSUV Model::AutomotiveValueTypes]	warning	UntDimEql	If a value is present for the unit attribute, the dimension attribute must be equal to the dimension property of the referenced unit.
Capacity [HSUV Model::HSUVRequirements::HSUVSpecification]	warning	RqAbst	The property isAbstract must be set to true.