Using Model-Driven Engineering to Support the Certification of Safety-Critical Systems

by

Rajwinder Kaur Panesar-Walawege

Thesis submitted for the degree of Philosophiae Doctor
Department of Informatics
Faculty of Mathematics and Natural Sciences
University of Oslo
July 2012

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo
No. 1212

ISSN 1501-7710

Cover: Inger Sandved Anfinsen.
Printed in Norway: AIT Oslo AS.

Produced in co-operation with Akademika publishing.
The thesis is produced by Unipub merely in connection with the
thesis defence. Kindly direct all inquiries regarding the thesis to the copyright
holder or the unit which grants the doctorate.

Preface

Discovery consists of seeing what everybody has seen and thinking what nobody has thought.

– Albert von Szent-Györgyi

Abstract

Critical systems such as those found in the avionics, automotive, maritime, and energy domains are often subject to a formal process known as certification. The goal of certification is to ensure that such systems will operate safely in the presence of known hazards, and without posing undue risks to the users, the public, or the environment.

Certification bodies examine such systems based on evidence that the system suppliers provide, to ensure that the relevant safety risks have been sufficiently mitigated. Typically, generic safety standards set forth the general evidence requirements across different industry sectors, and then derived standards specialize the generic standards according to the needs of a specific industry sector.

Regardless of whether a generic or sector-specific standard is being used, a key prerequisite for effective collection of evidence is that the supplier be aware of the requirements stipulated in the relevant standard and the evidence they require. This often proves to be a very challenging task because of the sheer size of the standards and the fact that the textual standards are amenable to subjective interpretation. Notably, suppliers find it hard to interpret the evidence requirements imposed by the safety standards within the domain of application; little support exists for recording, querying, and reporting evidence in a structured manner; and there is a general absence of guidelines on how the collected evidence supports the safety objectives.

This thesis proposes the application of Model-Driven Engineering as an enabler for performing the various tasks related to safety evidence management. The position taken is that models should serve as the main source of certification information - documents, when needed, should be generated from models. Models are beneficial for the purpose of safety certification in many respects, most notably: (1) Models can be employed to clarify the expectations of safety standards and recommended practices, and develop concrete guidelines for system suppliers; (2) Models expressed in standard notations avoid the ambiguity and redundancy problems associated with text-based documentation; (3) Models provide an ideal vehicle for preserving traceability and the chain of evidence between hazards, requirements, design elements, implementation, and test cases; (4) Models can represent different levels of abstraction and an explicit mapping between the different levels; (5) Models present opportunities for partial or full automation of many laborious safety analysis tasks.

The main contribution of this thesis is a model-driven process that enables the automated verification of compliance to standards based on evidence. Specifically, a UML profile is created, based on a conceptual model of a given standard, which provides a succinct and explicit interpretation of the underlying standard. The profile is augmented with constraints that help system suppliers with establishing a relationship between the concepts in the safety standard of interest and the concepts in the application domain. This in turn enables suppliers to demonstrate how their system development artifacts achieve compliance to the standard.

Additionally, UML profiles are further used to systematically capture how the evidence requirements of a generic standard are specialized in a particular domain. This provides a means of explicitly showing the relationship between a generic and a sector-specific standard. This tackles the certification issues that arise from poorly-stated or implicit relationships between a generic standards and their sector-specific interpretations.

Finally, the tool infrastructure needs for supporting the collection and management of

safety evidence data is tackled by proposing tools for upfront planning of evidence collection activities and the storage of evidence information outside of modelling environments.

Acknowledgements

This journey started in the fall of 2006 in Ottawa, when I attended a course given by Dr. Lionel Briand. The following year, when Dr. Briand moved to Norway, he presented me with the opportunity to join him at Simula Research Laboratory. Accepting that offer was an excellent decision.

I have been fortunate to have Dr. Briand and Dr. Sabetzadeh as my supervisors, their insightful guidance and support has been invaluable. I have learned from them how to do research and be a researcher - they have provided both scientific and practical advise. I have learned a lot from them and I am grateful to have worked with them.

Simula Research Laboratory and Simula School of Research and Innovation have provided excellent working conditions and I thank them for all their support over the years. I have been privileged to interact with people from different backgrounds and countries whilst at Simula and have thoroughly enjoyed my time there.

I have also had the opportunity to work with some wonderful colleagues: thanks to Hadi Hemmati and Shaukat Ali for great academic collaboration and fun times otherwise; Shaukat, Hadi, Aiko Fallas Yamashita and later Tao Yue and Zohaib Iqbal have been wonderful friends and the support network that every Ph.D candidate needs.

Finally, I would like to thank my parents and family for their support and patience while I pursued this milestone in my life. Last, but not least, none of this would have been possible without the love and support given to me by my spouse, Suneth Walawege. Before I ever contemplated of a Ph.d, Suneth told me that I was capable of doing one and always encouraged me to pursue it. Suneth, it was your belief in me that gave me the confidence to start and then complete this work. This is an achievement that I owe to you.

Rajwinder K. Panesar-Walawege, July 2012

Contents

List of Papers

- **Paper I**

Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand, Thierry Coq
Third IEEE International Conference on Software Testing, Verification and Validation (ICST), 2010

- **Paper II**

Using UML Profiles for Sector-Specific Tailoring of Safety Evidence Information
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand
30th ACM International Conference on Conceptual Modeling (ER), 2011

- **Paper III**

A Model-Driven Engineering Approach to Support the Verification of Compliance to Safety Standards
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand
22th IEEE International Symposium on Software Reliability Engineering (ISSRE), 2011

- **Paper IV**

Planning for Safety Evidence Collection: A Tool-Supported Approach Based on Modeling of Standards Compliance Information
Davide Falessi, Mehrdad Sabetzadeh, Lionel Briand, Emanuele Turella, Thierry Coq, Rajwinder Panesar-Walawege
IEEE Software (to appear)

- **Paper V**

CRESCO: Construction of Evidence Repositories for Managing Standards Compliance
Rajwinder Kaur Panesar-Walawege, Torbjørn Skyberg Knutsen, Mehrdad Sabetzadeh, Lionel Briand
Tool Demonstration paper at the 30th ACM International Conference on Conceptual Modeling (ER), 2011

- **Paper VI**

Using Model-Driven Engineering for Managing Safety Evidence: Challenges, Vision and Experience
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand
1st International Workshop on Software Certification at the 22th IEEE International Symposium on Software Reliability Engineering (ISSRE), 2011

- **Paper VII**

Supporting the Verification of Compliance to Safety Standards via Model-Driven Engineering: Approach, Tool-Support and Empirical Validation
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand
Submitted to the Journal of Information and Software Technology (IST),2012

Summary

1 Introduction

Increasingly, systems incorporating both software and hardware are used to automate highly complex tasks. Nuclear plants, oil production systems, airliners and automobiles are examples of such systems. As failures in these systems can cause harm to people or the environment, it is becoming crucial for these systems to be certified as safe for operation. A system is considered safe when it can perform its intended function without posing undue harm to its operators, the public, or the environment within which it operates. It is becoming the norm for safety critical systems to be certified by third-party certification bodies. The aim of certification is to provide assurance that the system has been deemed safe for use in a specific environment.

A key requirement of safety certification is the provision of evidence that a system complies with the applicable safety standards. The two main aspects under consideration here are: the standards involved, and the evidence that shows the compliance of the system with the specified standards. A common practice in defining standards for certification is to have a generic standard and then derive from it sector-specific standards for every industry sector that the generic standard applies to. The idea behind such a tiered approach is to unify the commonalities across different sectors into the generic standard, and then *specialize* the generic standard according to contextual needs.

The standards prescribe the procedures for compliance and system suppliers create the necessary evidence to meet the compliance requirements. Showing compliance to safety standards proves to be a very challenging task due to the fact that these standard are presented as very large textual documents that are amenable to subjective interpretation. If a derived standard is being used then there is the additional issue of understanding how the evidence requirements stated in a generic standard map onto those stated in a derived standard. The lack of a consistent interpretation can lead to misunderstandings about what evidence needs to be created - crucial evidence that should have been created during system development can be missed or unnecessary evidence artifacts may be created. This results in suppliers have to re-construct the required evidence at the time of certification - an often expensive and time consuming endeavour. On the certifier's side, poorly-structured and incomplete evidence often leads to significant delays and loss of productivity, and furthermore, does not allow the certifier to develop enough trust in the system undergoing certification. It is therefore very important to devise approaches, which are supported by effective automation, to specify, manage, and analyse the safety evidence necessary to demonstrate compliance to standards.

Finally, the evidence presented to the certification body needs to be highly structured in

order to ensure that it is readable and assessable. In general, there is a large amount of evidence that is gathered and all of it needs to be structured such that each piece of evidence and how it relates to other artifacts is clear to the certifier.

This thesis tackles these issues and presents coherent solutions that help suppliers of safety-critical systems to prepare adequately for certification. The solutions presented in this thesis use Model-Driven Engineering techniques [12] in a novel manner, showing that models can be used not only for system development but also for managing the evidence needed for certification.

Contributions

The contributions of this thesis are all related to effectively using certification standards for the certification of safety-critical systems. The basis of this work is an explicit interpretation of textual standards in order to promote a common understanding. We use conceptual models to create this explicit interpretation and to formalize the evidence requirements of a certification standard. The main contribution of this thesis is an approach that uses models to assist suppliers in managing the evidence required for certification according to a specific industry standard. Along side this work, three other direction for using conceptual models are explored: (1) how to make the relationship between a generic and derived standard explicit; (2) how to use conceptual models to guide planning for certification and (3) how to automatically generate an evidence repository.

There are seven papers included in this thesis. For Papers 1, 2, 3, 6 and 7, I was responsible for the idea, the implementation and writing. My supervisors contributed to all phases of the work with general advice and suggestions of improvements. Paper 4 was a tool paper implementing one facet of my work, I was responsible for the idea and the implementation was carried out by E. Turella under the guidance of D. Falessi with major contribution and feedback from myself and my supervisors. Paper 5 is another tool paper implementing another facet of this work where an evidence repository is generated from a conceptual model of a standard. The implementation was carried out by T. Skyberg Knutsen under supervision of myself and M. Sabetzadeh with major contributions from myself and L. Briand, I was responsible for writing this paper.

All the work presented in this thesis was carried out in collaboration with partners from industry. Both the suppliers and certifiers of safety-critical system were involved. The problems faced by both parties motivated the work and their needs were considered when creating the appropriate solutions.

Thesis Structure

This thesis is organized into two main sections:

Summary. This part explains the research conducted for this thesis and introduces the included papers. In Section 2, background information on the main concepts discussed in this thesis is presented and Section 3 describes the research goals of the thesis. Section 4 presents the research method employed, including the data collection procedures. A summary of the main results is in Section 5, while Section 6 discusses the future direction for this research. Section 7 concludes.

Papers. The rest of the thesis consists of the seven papers included in this thesis. The first six papers have been published in peer-reviewed conferences. The final paper has been submitted to an international journal. An overview of these papers is presented in Section 5.

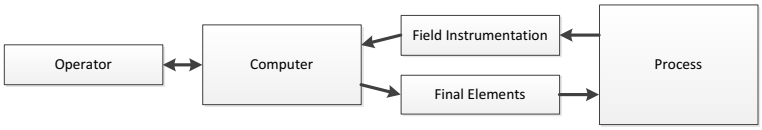


Figure 1: Components of a Safety-Critical System
[9]

2 Background

The main topic of this thesis is to present an approach that aids suppliers of safety-critical systems to prepare for certification in a systematic and structured manner. Before presenting the research undertaken and its results, it is necessary to define the concepts under study to ensure a common understanding. This section presents the concepts related to safety-critical systems and model-driven engineering.

2.1 Safety-Critical Systems

A safety-critical system is one in which the failure of the system can lead to injury or death of people and/or damage to property and the environment [9]. The basic form of current safety-critical systems that incorporate both hardware and software can be represented as shown in Figure 1 [9].

A safety-critical system is usually used to control some process, this could be the machinery for operating a drill in the sea-bed to extract oil, or it could be a control system for a car assembly line. There is a computer based system that monitors and controls the process. The monitoring is done via the field instrumentation in the form of various types of sensors such as temperature sensors or pressure sensors. The sensors send information of the process being controlled to the computer system. The computer system has some software application running that makes decisions based upon the received input. When the computer system needs to affect a change in the process being controlled, it does so via some type of actuators, sometimes called final elements. The final elements are used to affect a change in the physical parameters of the process under control, e.g. open a valve to release pressure. The operator is the human presence in this system, who oversees and manages the function of the overall system.

2.2 Safety Certification

Safety-critical systems are typically subject to a safety certification process by licensing and safety regulatory bodies. The aim of certification is to provide assurance that a system has been deemed safe for use in a specific environment. This is done by setting safety objectives for the system being constructed. The safety objectives are based on the potential hazards that can occur during the operation of the system. Suppliers are then required to demonstrate how their system meets the safety objectives. Demonstrating the satisfaction of the safety objectives involves gathering evidence during the lifecycle of a system and constructing well-reasoned arguments that relate the evidence to the objectives. The evidence is then presented to a third-party certification body that assesses the system and issues a certificate,

permitting the system to be deployed.

In order to regulate the certification of safety-critical systems, there are industry recognized certification standards that set out the requirements for creating safe systems. The suppliers follow the standards to create the evidence for certification. Thus it is important that there is a consistent understanding and an agreed-upon interpretation of the standard being used, and all parties involved (supplier and certifier) should know what evidence is to be collected and maintained in readiness for certification.

2.3 Standards for certification

The certification standards used for the certification process can be in the form of either national or international standards. Often, the requirement to adhere to a particular standard may be part of a contract of purchase for a system. The use of a standard gives the purchaser of a system some idea of the process that was followed to construct the system and also confidence in its use - since it comes with a certificate from a regulation body stating that it is safe for operation. In some cases the certificate is also used in legal issues when an accident has occurred - the owner of the system can use the certificate to show that the system had been approved for operation, or the supplier can use it to show that the system had satisfied all the requirements of the requisite standard when it was constructed.

Standards can be either generic standards that apply to a specific technology used in a number of diverse domains or they can be domain specific for a particular technology. For example, there is the general standard IEC61508 [16] which is used for the certification of electrical, electronic or programmable electronic systems that are used in safety-critical environments. This standard has been written such that it can be used stand-alone, or customised for a specific domain. The IEC61511 is one specific customization of this standard for the process industry [15]. There are domain specific standards that do not originate from generic standards, such as the DOD 178B standard that is specifically used for the certification of software used in airborne systems and equipment [23].

As control systems are being used to control increasingly complex systems, it is becoming the norm to certify safety-critical systems based upon industry relevant standards. Thus, making effective use of standards is an important issue for all parties involved in the construction and operation of safety-critical systems. Some of the issues faced when using standards will be highlighted in Section 3. In this thesis, the main standard used is the IEC61508 standard and the next section provides some background information on this standard. The work in this thesis is based on the 1998 edition of the standard, however, since then a new version has been released.

2.3.1 The IEC61508 standard

IEC61508 is concerned with improving the development of safety-related electrical/electronic/-programmable electronic systems (E/E/PES) whose failure could result in harm to people, equipment, and/or the environment. In this standard, the term safety-related system is used and refers to safety-critical systems. As stated earlier, the IEC61508 is a generic standard and can either be used directly or for the creation of domain-specific standards in industries that require an equivalent level of safety.

The standard applies to both low-demand and continuous mode systems. In a low-demand system, the frequency of demands for operation is low (the standard specifies a precise range). An example of a low-demand system is a fire & gas protection system, which alerts personnel if a fire or gas leakage is detected and initiates protective actions either automatically or through manual intervention. A continuous (or high-demand) mode system is one with a high frequency of demands for operation. An example would be the dynamic positioning system that continuously controls a vessel's movement when the vessel is near a port or rig.

The goal of the standard is to ensure that safety-related E/E/PES systems operate correctly in response to their inputs in order to either achieve or maintain a safe state of the equipment under control. This is referred to as *functional safety*. Functional safety is not all there is to safety. For example, the activation of an alarm in response to a fire breakout is a functional safety measure, whereas the use of fire resistant walls to control the spread of fire is not, although the latter measure protects against the same hazard. IEC61508 deals only with functional safety. A function that a control system performs to ensure that the system remains in a safe state is referred to as a safety function. Each safety function specifies what safety objective is to be achieved (safety function requirement) and the level of integrity with which the safety function is implemented, known as safety integrity level (SIL). The standard specifies four different safety integrity levels based on the consequences of the failure of the system. Level four is the highest level and assigned to systems whose failure could have a catastrophic impact on people and the environment. Level one is the lowest level and assigned to those systems whose failure only has minor damage to property.

The IEC61508 standard consists of seven parts, part zero explains the concepts of functional safety and provides an overview of the other parts that make up the standard. Part one contains general requirements for the functional safety of E/E/PES systems and part two has specific requirements for the hardware components whereas part three concerns the software employed in E/E/PES systems. Part four contains definition and explanations of any abbreviations used in the standard and part five contains examples of methods for the determination of safety integrity levels. Part six contains guidelines on how to apply parts two and three and part seven is an overview of the measures and techniques used to show the different levels of safety that have been achieved. Thus the actual requirements for functional safety are in parts one, two and three. To systematically deal with the activities necessary to achieve the required level of safety (safety integrity level), each of these sections prescribes a safety lifecycle, part one for the overall system, part two for the hardware components and part three for the software components. The overall safety lifecycle is shown in Figure 2. The purpose of each phase of the lifecycle is described below, noting that phases 10 and 11 are outside the scope of the standard:

1. ***Concept:*** To develop a sufficient level of understanding of the Equipment Under Control (EUC) and its environment in order to enable the other safety lifecycle activities to be satisfactorily performed.

2. ***Overall scope definition:*** To determine the boundary of the EUC and the EUC control system; to specify the scope of the hazard and risk analysis.

3. ***Hazard and risk analysis:*** To determine the hazards and hazardous events of the EUC and the EUC control system; to determine the event sequences leading to the hazardous events; to determine the EUC risks associated with the hazardous events.

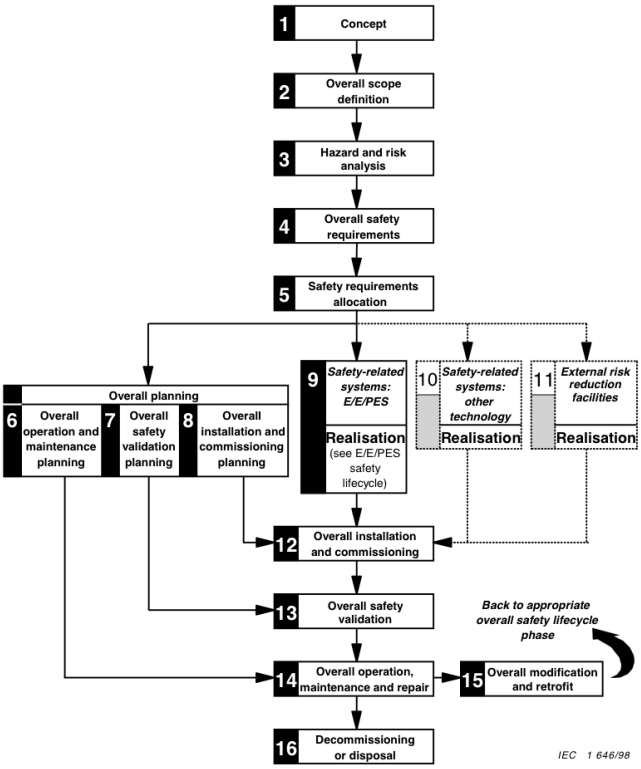


Figure 2: Safety Lifecycle in the IEC61508 standard [16]

4. ***Overall safety requirements:*** To develop the specification for the overall safety requirements necessary to achieve the required functional safety.

5. ***Safety requirements allocation:*** To allocate the safety functions in the overall safety requirements specification to the designated E/E/PE safety-related systems, other technology safety-related systems, and external risk reduction facilities; to allocate a safety integrity level to each safety function.

6. ***Overall operation and maintenance planning:*** To develop a plan for operating and maintaining the E/E/PE safety-related systems, in order to ensure that the required functional safety is maintained during operation and maintenance.

7. ***Overall safety validation planning:*** To develop a plan to facilitate the overall safety validation of the E/E/PE safety-related systems.

8. ***Overall installation and commissioning planning:*** To develop a plan for the installation of the E/E/PE safety-related systems; to develop a plan for the commissioning of the E/E/PE safety-related systems.

9. ***E/E/PE safety-related systems realization:*** To create E/E/PE safety-related systems conforming to the specification for the E/E/PES safety requirements.

10. ***Other technology safety-related systems realization:*** To create other technology safety-related systems to meet the safety function requirements at the integrity levels specified for such systems.

11. ***External risk reduction facilities realization:*** To create external risk reduction facilities to meet the safety function requirements at the safety integrity levels specified for such facilities.

12. ***Overall installation and commissioning:*** To install and commission the E/E/PE safety-related systems.

13. ***Overall safety validation:*** To validate that the E/E/PE safety-related systems meet the specification for the overall safety requirements.

14. ***Overall operation, maintenance, and repair:*** To operate, maintain and repair the E/E/PE safety-related systems in a way that the required functional safety is maintained.

15. ***Overall modification and retrofit:*** To ensure that the functional safety for the E/E/PE safety-related systems is appropriate, both during and after modification and retrofit.

16. ***Decommissioning or disposal:*** To ensure that the functional safety for the E/E/PE safety-related systems is appropriate during and after decommissioning or disposing of the EUC.

For clarity, activities relating to the ***management of functional safety*** (i.e., specifying the management and technical activities during each phase, and responsibilities of those who are to carry out the phase), ***verification*** (i.e., demonstrating that the outputs of each phase meet the objectives and requirements of that phase), and ***functional safety assessment*** (i.e., investigating and forming a judgment on the functional safety achieved) are not shown in Figure 2. These activities can be potentially relevant to all lifecycle phases and need to be applied wherever necessary.

Phase 9 of the overall safety lifecycle is concerned with the realization of E/E/PE part of the system. This phase is expanded and shown in Figure 3 (for details of the activities in this phase, see Part 2 of the standard). The standard requires an explicit safety lifecycle, shown in Figure 4, for the software deployed on a PES. Thus both the hardware and the software have their own development lifecycles, following by phases of integration of the two.

In order to comply with the standard, suppliers have to show that all hazards in the system have been identified and mitigated according to the safety integrity level assigned. They also have to show that they followed the development lifecycle using the techniques recommended by the standard and that competent professionals carried out the work.

2.4 Model-Driven Engineering

This thesis purports the use of models as the foundation of the work presented. Models have been the basis of all traditional engineering disciplines, where constructing models to

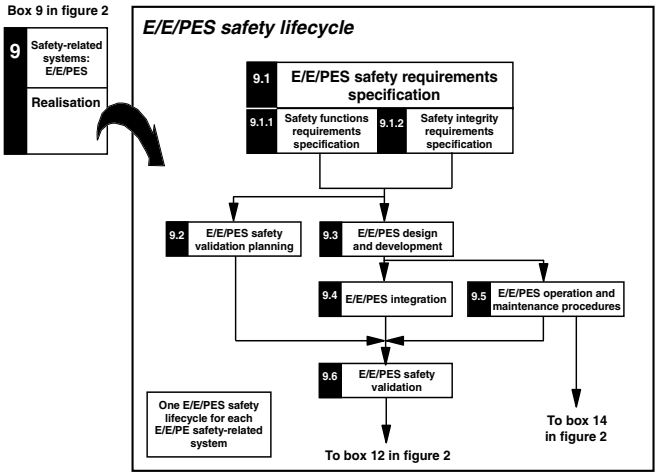


Figure 3: IEC 61508 E/E/PES safety lifecycle (in realization phase) [16]

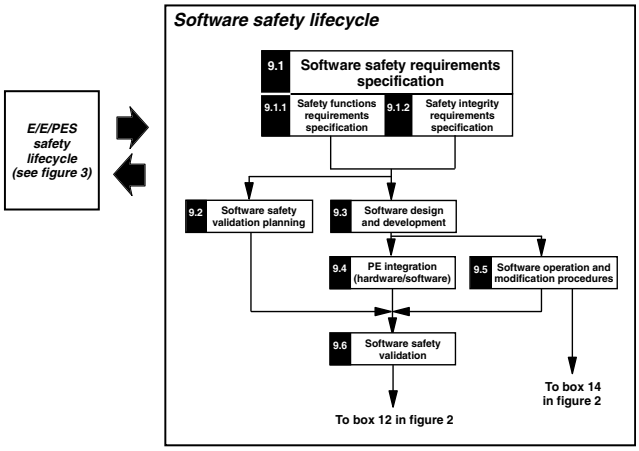


Figure 4: IEC 61508 Software safety lifecycle (in realization phase) [16]

understand and solve complex problems is the norm. Models provide a means of abstracting away any irrelevant details and focusing on a specific viewpoint thus helping to alleviate complexity. As software is being used in increasingly complex contexts, it is no longer possible to understand the complexity involved without creating useful abstractions of the system - to both, understand the domain of the system and to create the necessary solutions.

In the context of software engineering, Model-Driven Engineering (MDE), refers to the use of models and model transformations, at various level of abstraction, for the creation of software systems [12]. Gašević et al. characterize MDE by two main relations - representation and conformance. Representation means that a model represents a software artifact or a real world domain. Conformance means that a model must conform to a higher level model that describes it - a metamodel. A metamodel is described as the set of constructs and rules needed to build specific models within a particular domain of interest.

In choosing a modelling language for the work carried out in this thesis, there were a number of issues to consider:

(i) An existing, well known modelling language had to be employed. This was important to the safety-critical system suppliers. They are sceptical about adopting new untried tools and languages into their development practices as they have to justify the use of any new tool and language.

(ii) The modelling language had to be able to model very diverse types of information. There were three requirements here: (1) the need to model a safety standard; (2) the need to create a domain model of the system, which in this context is all the physical and abstract components of a family (class) of systems in a particular application area, the environment in which this family of systems functions, and the key artifacts built throughout its development; and (3) the need to provide a means of linking the two types of models together to show how the system satisfies the standard.

(iii) The creation of the new models had to integrate easily with the development processes already being employed by the suppliers. This was important in order to ensure that the creation of the models did not impact the productivity of the development team - this point is important in improving the chances of adoption of a new approach.

The main choice was between using a general-purpose modelling language(GPML) or a domain-specific modelling language (DSML).

2.4.1 Unified Modeling Language versus Domain-Specific Modelling Languages

The most commonly used GPML in software engineering is the Unified Modeling Language (UML) [22]. It is a standardized modelling language developed by the Object Management Group (OMG) [2] and conforms to their metamodel that is expressed in the Meta-Object Facility (MOF) [3]. UML is used for the specification, construction and documentation of software artifacts. It provides a means for clearly and precisely communicating information regarding the artifacts of software. UML provides modelling constructs that allow for modelling the static structure of a system as well its behaviour at different levels of abstraction. All the constructs can be visualized via a number of different diagrams. The static structure diagrams

are class diagrams, object diagrams, component diagrams and deployment diagrams whereas behaviour is presented via activity diagrams, sequence diagrams and state machine diagrams.

UML has a well defined syntax and semantics. The syntax is divided into two kinds: the abstract syntax that defines the kinds of elements and how they relate to other elements and the concrete syntax that defines what the elements look like. The semantics of UML are expressed using a combination of formal statements in the Object Constraint Language (OCL) [1] and informal text. UML is designed to model the diversity that exists across the different application domains that it can be used in. To allow for this diversity, UML has necessarily been defined with semantic variation points - these are points of variation in the semantics of the metamodel that provide an intentional degree of freedom for the interpretation of its semantics [7]. These semantic variation points may be defined differently in different domains and are usually of little concern for high-level modelling of systems, however, at a certain level of detail, there will be a need to use some mechanism to provide the required domain-specific interpretation [19]. The UML specification provides extensibility mechanisms for this purpose in the form of UML profiles.

Domain-specific modelling languages, on the other hand, are custom built languages that purposely target a specific domain. An advantage of DSMLs is that they allow the end-users to define the system according to the needs and perspective of the user domain whilst shielding them from any implementation details. Model transformations are then used to translate the domain-specific models into either further refined models, or as is most often the case, into automatically generated source code.

A DSML requires a meta-modelling infrastructure to be in place that will aid in the creation of the abstract and concrete syntaxes as well as the semantics. A common infrastructure for creating the abstract syntax is the Eclipse Modelling Framework [10] which employs the Ecore metamodel for defining the abstract syntax of a DSML. Further tools exist to create the concrete syntax in either a graphical format or a textual one. The semantics of the language also have to be agreed upon and appropriately defined using a suitable notation. Note that this as important challenge in the adoption of DSMLs - often it is the syntax that is formally defined and not the semantics [5, 6, 18]. DSMLs are meant to be easier to learn by the domain experts, however their creation requires expert knowledge not only in the domain of use, but also in the design of languages and well as expertise in the existing tools available for their creation. This is a challenge as the infrastructure available for their creation is not yet mature [5].

2.4.2 Modelling Language Selection

Using a DSML for creating models is useful when we have a specifically defined domain. The work in this thesis involves modelling safety standards as well as systems. Regarding the standard, the information in the standard is being modelled to create an explicit interpretation. The contents of the standard are being modelled not its structure - thus we are not dealing with sections, paragraphs, numbered requirements and so forth but with the actual concepts expressed in the standard. These vary greatly from expressing hazards and risk, to development artifacts and system components. The standard itself is applicable to multiple domains from safe load indicators on a crane to railway signalling systems and fly-by-wire operations of aircraft flight controls. Hence, there are no specifics, not in the case of the standard nor in the

type of system to be modelled. The types of models to be created are quite diverse and are best expressed using a general-purpose modelling language.

The models are being created to show how the system satisfies the requirements of the standard. Thus there is a need to link the two models together and yet maintain a clean separation between the two so that the concepts of one do not confuse those of the other. This is better achieved by using UML and its extension mechanism of profiles as opposed to defining two DSMLs - one for the standard and one for the system. Which still leaves the problem of how to link the two types of models together whilst providing guidance in doing so.

An advantage of using DSMLs is that source code can be automatically generated from the models, this too is not applicable in the context of this work, where the models are created for certification and not code generation.

After deliberating on the types of constructs that will be needed for creating the models, it was agreed that constructs to captures concepts, attributes, relationships, packages, enumerations, and inheritance would be sufficient for modelling the standard and the domain models of the system. These concepts are present in both DSMLs as well as UML. It seemed a sensible choice to use a language that already has those constructs, instead of spending time re-inventing them again.

UML, as a general-purpose language fits all the identified needs:

- It is a standardized language that is used within industry and is well accepted as a general purpose modelling language.
- The model of the certification standard and the domain model of the system can be created using the same language. It is also more likely that a UML model of a system (perhaps partial) will exist that may be used as a basis for the certification model.
- The UML profile provides a clear and clean link between a model of a system and the model of a standard and OCL constraints can be used to provide guidance in linking the two models together.

Furthermore, the use of UML has been specifically limited to class diagrams to ensure that too much complexity is not introduced. UML class diagrams are easy to understand and hence their choice. The other constructs of UML have not been used in the interest of keeping things simple and thus easy to adopt in industry, whilst still leaving the option of being able to extend this work to include other constructs in the future. The constructs not being used do not pose an issue, the users are informed that the profile works only with the constructs used in class diagrams and not other UML constructs. Should they want to use other constructs, the profile needs to be extended. However, they are free to use other models for any other analyses that they want, the use of the profile will not hinder them either. The only constraint is that the certification constraints are checked on class diagrams only.

Although, UML is criticised for not having formal semantics and allowing variation points, this is only an issue if there is a need to mathematically analyse the information in the models. In our case we use OCL to add the semantic information that is required and then validate the OCL constraints, hence for the purpose of this work, the lack of formal semantics or the presence of semantic variation points is not really an issue.

The focus of the thesis is to use models, and visualize the models as and when necessary. UML provides this capability: the models can be created textually or using the graphical notation provided. To aid with the exemplification of the work, the thesis focuses on the graphical notation but this is by no means a necessity when using UML. The tool infrastructure used for creating the UML models is Rational Software Architect [13] which specifically provides a mechanism for automatically generating the graphical representation of the models when required - these are called browse diagrams. This is a useful facility when creating very large models, it is not possible nor useful to visualize the entire model, but it is useful at times to visualise parts of it. Thus the ability of UML to provide both a textual and graphical notation is helpful indeed. In this thesis class diagrams are used for visualizing the static structure of either systems (domain models of systems are created) or concepts retrieved from standards (conceptual models of standards are created). Object diagrams are used for visualizing instantiations of the class diagrams.

UML allows the creation of the conceptual model of the standard which along with a glossary provides a means for creating an explicit interpretation of the modelled concepts. The conceptual model is then converted into a UML profile that is applied to domain models of the system. This extension mechanism is one means of adapting UML to a specific domain, however in this work it is used to provide a clean and clear connection between the certification standard and the domain models of the system to be certified. The UML profile is used adapt UML to the terminology used in a specific certification standard. The next section discusses how profiles are created and used.

2.4.3 UML Profiles: Extension Mechanism of UML

The UML standard provide a mechanism for extending the definition of its concepts so that they can be customized for use in diverse domains or platforms. This mechanism is the creation of a UML profile. A UML profile defines stereotypes which are extensions of the elements of the UML metamodel. Stereotype are used to introduce new domain-specific terminology and attributes to model elements. A simple example is shown in Figure 5.

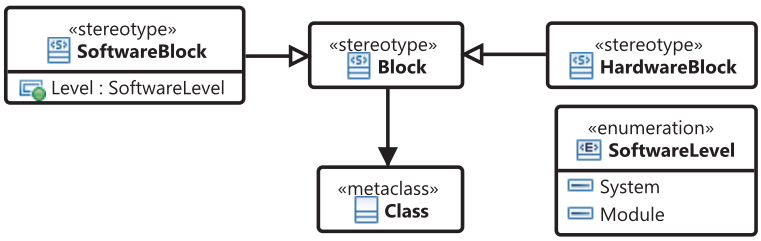


Figure 5: Stereotype Example

The UML metaclass `Class` is being extended (indicated by the filled in arrow from `Block` to `Class`). The stereotypes defined are `Block` and its specializations `SoftwareBlock` and `HardwareBlock`. The stereotype `SoftwareBlock` has an attribute named 'Level' that is used to state whether the software is at the system level or the module level as shown by the enumeration type `SoftwareLevel`. As this stereotype extends the metaclass `Class`, it can

be applied to instances of this metaclass - the `Class` construct in UML class diagrams, an example is shown in Figure 6.

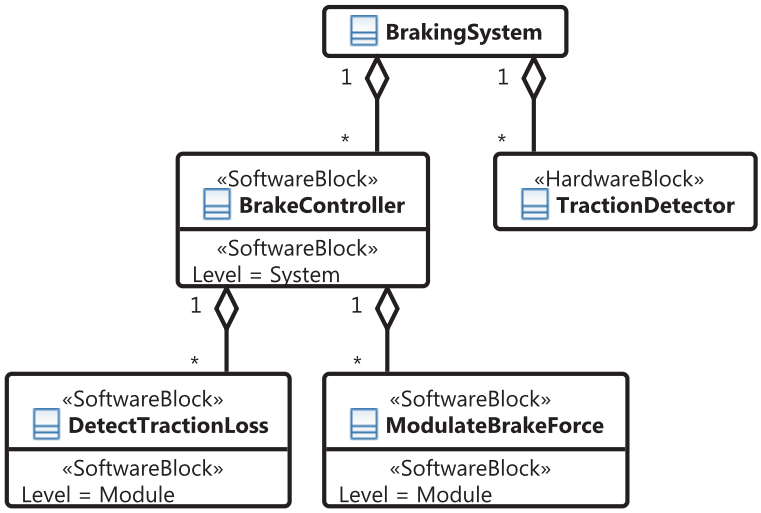


Figure 6: Stereotype Applied to 'Class' Elements in a Class Diagram

A very simplified view of the braking system for a car is shown, consisting of a brake controller and a traction detector. The brake controller is software that is made up of two modules - one for detecting loss of traction and one for applying the brakes. In this example, we can use the stereotypes to show which part of the system is hardware and which is software and for the software we can also use the stereotype attribute to show the level of the software - the system software is the `BrakeController` and the two modules it consists of are `DetectTractionLoss` and `ModulateBrakeForce`.

Sometimes it may be necessary to ensure that elements with certain stereotypes also have certain other properties. This can be done by defining OCL [1] constraints on the stereotypes. OCL is a standardized language for writing constraints based on first order logic. The constraints can be added to various types of model element. In this thesis, constraints are added to stereotypes and then checked on the elements to which the stereotypes are applied.

3 Research Goals

Safety-critical systems are certified based on industry relevant standards which set out the requirements that must be fulfilled to gain certification. Without standards upon which to base the certification requirements, the process would become ad hoc. Standards provide a means of accumulating and sharing best practices and providing a structure to the certification process. However, the use of standards does pose some challenges to system suppliers and certifiers alike. This section details these challenges and the related research goals for this thesis.

3.1 Goal 1: Creating common interpretations of certification standards.

Suppliers of safety-critical systems are well aware of the safety standards to which their systems will be certified. In preparation for certification, they need to have the requisite evidence ready. To create this evidence, they need to read and understand the relevant standard and the requirements therein. In order to use the standard effectively, there needs to be a consistent understanding and agreed-upon interpretation of the standard being used, and all parties involved should know what evidence is to be collected and maintained in readiness for certification. This common understanding also needs to extend to the certification body – it is hardly useful for the supplier and the certifier to have different interpretations of the standard being used. The different interpretations occur due to the standards being rather large documents expressed textually in a language not easily understood by everyone in the organization. Thus, they are amenable to subjective interpretation. This is an issue well recognized in the literature, Redmill [24] addresses these issues in the context of IEC61508 standard, where readers have difficulty understanding the standard and engineers are unable to interpret the standard consistently throughout an organization. Feldt et. al [11] discuss it regarding standards used in the space industry, where there have been problems between customers and suppliers due to the differences that exists in the interpretation of the applicable standards. Most recently, Sannier et. al [26] have found the same problem in the nuclear energy industry where there are gaps between the possible interpretations of the same standard. Thus, there is a need to have a common and formal interpretation of the requirements of a standard on which certification can be based.

3.2 Goal 2: Specializing standards to industrial contexts.

As stated in Section 1, standards may be adapted to the context of their use such that there is a generic standard and its sector-specific derivations. This practice is identified as *specialization* in this thesis. IEC61508 is one such standard that has been adapted to a number of different sectors. In the process industry, this standard is adapted as IEC61511 [15], in railways as EN50128 [14], in the petroleum industry as OLF070 [27], and in the automotive industry as the forthcoming ISO 26262 [17].

To effectively use derived standards, it is important to know which requirements of the generic standard map on to the sector-specific standard. This specification of the relationship between two standards can also be necessary between two standards within the same industry sector. Sometimes standards within a sector evolve, leading to different systems being

specified to different versions of the same standard. In this case we again have a parent standard and another that is derived from it, and we still have the issue of systematically specifying the relationship between the two. Feldt et. al. [11] cite the lack of agreed-upon relationships between generic and derived standards as one of the main reasons behind certification delays within the space industry. Whatever the case, there has been little work to date on systematizing the specification of the relationship between generic and derived standards. Furthermore, Nordland [20] notes the lack of a well-formulated process for showing that a derived standard is consistent with a generic one. This too is directly attributable to the lack of precise and explicitly-defined relationships between the standards.

3.3 Goal 3: Aligning Standards to organizational practices.

When a standard is being used within an organization it will need to be aligned to the practices of the organization. In this manner, the organization can check which of its existing practices comply with the standard and which new practices need to be introduced and tailored. Thus, there is a need to assist system suppliers in relating the concepts of their application domain to the evidence requirements of the applicable standards. This observation is based on the fact that the majority of the evidence artifacts that the suppliers create and manage are based on the concepts for the application domain, as opposed to the concepts of the certification standards. The certification body also needs to interpret the standard in the context of the application domain in order to understand how the evidence relates to the standard before it can check whether sufficient evidence exists to satisfy all the requirements of the standard. This highlights the need for a systematic procedure for creating the necessary evidence and presenting it in a form that will allow the certification body to assess it in terms of both the application domain and the relevant standard.

3.4 Goal 4: Planning for certification.

Inherent in the three challenges above is the need to ensure that the supplier and the certifier are both using the same interpretation of the standard and that both have an upfront agreement concerning the evidence artifacts that will be created during system development. If no such agreement exists, then it is possible that the supplier may create evidence artifacts that do not match the certifiers' interpretation of the standard, or the supplier may be missing certain artifacts that the certifier deems necessary. This mismatch can be a costly affair for the supplier who will need to remedy the situation after the system has already been developed. Thus, there needs to be a systematic procedure for ensuring an upfront agreement regarding the specifics of the evidence artifacts.

3.5 Goal 5: Managing safety evidence electronically.

The final form in which the evidence is presented for certification needs to be highly structured in order to ensure that it is readable and assessable. In general, there is a large amount of evidence that is gathered and all of it needs to be structured such that each piece of evidence and how it relates to other artifacts is clear to the certifier. Traditionally, this has been very difficult to achieve via the paper-based documents that form the basis of the certification

evidence. Thus, there is a case for managing this evidence electronically [8] in order to ease navigation of the information and to allow for diversity of presentation, delivery and re-use.

3.6 Goal 6: Promoting re-use of safety evidence.

The type of systems that are usually certified are characterized as belonging to product families that have many variants of a base system. Thus any proposed solution for managing certification evidence should take advantage of the re-use that is possible in these types of systems to create a systematic and cost-effective solution.

Given these goals, a means to deals with different levels of abstraction in the information that needs to be represented is needed: going from generic standards to specialized standards, from product family to a specific variant of a system and a way to explicitly define the relationships between the two. This, combined with the need to structure the information systematically and electronically, led to the conclusion that the use of models would be an ideal way to cover all the goals.

4 Research Method

The research method employed for this thesis is referred to as 'Design and Creation' [21]. This is a common method of conducting research whereby the focus is on developing new information technology (IT) products to solve a problem. The end product is often an instantiation of a computer-based system but it may also be a new construct, model or method. A new construct could be the introduction of new concepts or vocabulary for the domain, models are a combination of constructs that are used to help in the understanding of a problem and the development of a potential solution whereas a method would be some guidance on such models and the process employed to solve a problem using them. Of course, an instantiation is a working system that demonstrates how the constructs, models and methods are implemented in a computer-based system. The steps in this research method are awareness, suggestion, development, evaluation and conclusion.

4.1 Awareness:

This is the recognition of a problem, it is akin to finding which research problem to solve. The research problems to tackle within this thesis were based on the personal experience of the author, the needs of the industry partners, a survey of the literature and field research. The author spent three months at a partner company to learn about their systems and identify the issues that required solutions and would warrant research at the doctoral level. The field work entailed observing the work of the engineers involved in developing a safety-critical application and participating in the creation of a domain model of the system to help them achieve an overall picture of their system. Based on the discussion with practitioners, the six goals discussed in Section 3 were developed as the research goals for this thesis.

4.2 Suggestion:

This is the proposal of a solution that will solve the goals identified. For this thesis, a model-driven approach for certification is proposed. Models are used (1) to clarify the expectations of safety standards and recommended practices, and develop concrete guidelines for system suppliers; (2) to alleviate the ambiguity and redundancy problems associated with text-based documentation by using standard notations; (3) to provide a means for preserving traceability and the chain of evidence between hazards, requirements, design elements, implementation, and test cases; (4) to represent different levels of abstraction and an explicit mapping between the different levels; (5) to partially or fully automate some of the laborious safety analysis tasks (e.g., completeness and consistency checking). A number of solutions are put forth for solving the different problems identified in the goals of the thesis. In some cases new software was created (for goals 4 and 5) but in other cases (goals 1, 2, 3 and 6) new approaches are proposed that use models to guide suppliers in preparing for certification. Guidance is provided on which models to create how to create them as well as how to use them within the overall approaches that are proposed.

4.3 Development:

This is the step in which the proposed solutions are implemented. Within this thesis the main contribution was the creation of an approach to aid system suppliers in preparing for certification - a solution was implemented for sub-sea production systems [4] and the IEC61508 standard [16]. All the steps of the proposed approach were carried out by creating a conceptual model of the IEC61508 standard, followed by constructing a UML [22] profile of the same standard. The domain models for a part of a sub-sea production control system were created and used to demonstrate how the system development artifacts achieved compliance to the standard. For the other goals, two additional solutions were implemented one to aid planning for certification and the other to automatically construct evidence repositories.

4.4 Evaluation:

This is the process of analysing the developed IT products to assess their usefulness. In this thesis industry relevant research is presented and the proof of the usefulness of an approach in industry is its adoption for use by practitioners in the field. Thus for the main contribution of the thesis, which is the model-driven approach for preparing for certification, a pilot study was carried out to show the application of the approach in the maritime and energy domain, followed by a survey of domain experts to determine their perceptions regarding the approach.

4.5 Conclusion:

This is the final step where the results of the whole process are written up. This is the aim of this thesis. The research carried out in this thesis has resulted in two tools that help with the fulfilment of goals 4 and 5 of the thesis, an approach for maintaining consistency amongst generic and derived standards and a novel approach for using UML profiles and model-driven engineering to enable the automated verification of the compliance of safety-critical systems to industry relevant certification standards. For this approach, guidance is provided on the models to be produced and how these models are used within a process to aid certification. Existing tool support that is normally used within the context of system development is used in a novel way to aid in the preparation for the certification process. Furthermore, the approach has been applied in the maritime and energy domain showing its feasibility in an industrial context with acceptable effort involved. The majority of surveyed practitioners found the approach easy to understand and were in favour of adopting the approach in their work.

4.6 Data Generation Methods

Within the research strategy of design and creation, different types of data generation methods can be employed. The two main techniques used in this thesis are case study and survey. A case study focuses on investigating phenomena in their context [25]. A case study is suited when it is difficult to separate the phenomena under study from its context. In this thesis a case study was carried out to investigate the feasibility of the proposed model-driven approach for certification. In this case study, document analysis and a questionnaire were used to generate the data. Certification standards were analysed to create the conceptual model, books and other domains standards were used to create the domain models of the system and documents

detailing internal procedures of the supplier company were used to create the development process models. Subsequently a survey was carried out to gain insight into the perceptions of practitioners at the supplier company. A workshop was carried out during which all the steps of the approach were explained in detail to a group of industry participants; the results of the case study were presented and tool support was demonstrated. A question and answer session was carried out to clarify any concerns. This was followed by a questionnaire that participants answered to provide their perceptions. In this manner we were able to collate their perceptions about the presented approach.

Figure 7: Relationship between Research Goals and Research Papers.

5 Summary of Results

Seven research papers represent the main research results of this thesis. The relationship between the papers and the research goals is shown in Fig. 7. Paper I addresses research goal 1; paper 2 addresses research goal 2; paper III addresses research goals 3,4,5 and 6; paper IV addresses research goal 4; paper 5 addresses research goal 5 as well but looks into automatically creating repositories for evidence information. Paper VI is a vision paper that explains our overall strategy for managing certification evidence and paper VII is a journal extension of paper III and presents an empirical evaluation of the approach for specifying and analysing safety evidence in order to show conformance to a safety standard.

Paper I: Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard.
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand, Thierry Coq. Third IEEE International Conference on Software Testing, Verification and Validation (ICST), 2010.

This paper addresses the goal of creating an explicit interpretation of the requirements of a safety standard. This work was motivated by the fact that little has been done to precisely characterize what evidence should be collected to comply with a safety standard, leaving suppliers with little guidance on what evidence to collate during the development process. This has left the suppliers having to recover the relevant evidence after the fact – an extremely costly and sometimes impractical task. Although standards such as the IEC61508 – which is widely viewed as the best available generic standard for managing functional safety in software – provide some guidance for the collection of relevant safety and certification information,

this guidance is mostly textual, not expressed in a precise and structured form, and is not easy to specialize to context-specific needs. To address these issues, a conceptual model is presented to characterize the evidence for arguing about software safety. The model captures both the information requirements for demonstrating compliance with IEC61508 and the traceability links necessary to create a seamless chain of evidence. Furthermore, the paper explores initial ideas on how a conceptual model can be specialized according to the needs of a particular context, and discusses some important ways in which the model can facilitate software certification. Examples are presented to show how the conceptual model can be specialized according to the needs of a particular context and describes some important ways in which the model can facilitate software certification. An analysis of a random sample of issues raised in certification meetings shows that a majority of them would have been prevented or addressed by information collected according to the model. Applications of the model include: the precise specification of safety-relevant information requirements for system suppliers; definition of a data model for developing a certification repository; the implementation of automatic, safety-relevant constraint verification (e.g., compliance with standard, recommended practice); and the automated generation of certification reports on demand.

Paper II: Using UML Profiles for Sector-Specific Tailoring of Safety Evidence Information.
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand.
30th ACM International Conference on Conceptual Modeling (ER), 2011.

This paper addresses the goal of specializing standards to industrial contexts. A key requirement of certification is the provision of evidence that a system complies with the applicable standards. The way this is typically organized is to have a generic standard that sets forth the general evidence requirements across different industry sectors, and then to have a derived standard that specializes the generic standard according to the needs of a specific industry sector. To demonstrate standards compliance, one therefore needs to precisely specify how the evidence requirements of a sector-specific standard map onto those of the generic parent standard. Unfortunately, little research has been done to date on capturing the relationship between generic and sector-specific standards and a large fraction of the issues arising during certification can be traced to poorly-stated or implicit relationships between a generic standard and its sector-specific interpretation. This paper presents a methodology for ensuring that a generic standard can be specialized in a systematic manner for a particular domain. This is done by capturing the generic standard as a conceptual model using a UML class diagram and using this as a basis for creating a UML profile. The profile is then applied to the conceptual model of a sector-specific standard and used as an explicit means of keeping track of the relationships between the two. The methodology is exemplified by showing excerpts of the IEC61508 conceptual model that was created earlier, the UML profile based on this model and how this profile is applied to a conceptual model of the OLF070 standard which is a sector-specific derivation of IEC61508 for the petroleum industry.

The approach offers two key benefits: (1) It incorporates the specific concepts used by a generic standard into the sector-specific standard whilst making a clear distinction between the two; and (2) It explicitly captures the mapping between two standards and defines consistency

rules between them, which can be automatically verified and used for providing guidance to the users about how to resolve any inconsistencies.

Paper III: A Model-Driven Engineering Approach to Support the Verification of Compliance to Safety Standards.
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand.
22th IEEE International Symposium on Software Reliability Engineering (ISSRE), 2011.

This paper addresses the goal of aligning standards to organizational practices and managing safety evidence electronically whilst promoting re-use. Certification of safety-critical systems according to well-recognised standards is the norm in many industries where the failure of such systems can harm people or the environment. Certification bodies examine such systems, based on evidence that the system suppliers provide, to ensure that the relevant safety risks have been sufficiently mitigated. The evidence is aimed at satisfying the requirements of the standards used for certification, and naturally a key prerequisite for effective collection of evidence, is that the supplier be aware of these requirements and the evidence they require. This often proves to be a very challenging task because of the sheer size of the standards and the fact that the textual standards are amenable to subjective interpretation.

This paper shows how to use model-driven engineering principles and technology to specify and analyze safety evidence in order to show conformance to a safety standard. This is done by establishing a sound relationship between a domain model of a safety-critical application and the evidence model of a certification standard. The requirements of the relevant standard are captured as a conceptual model using a UML class diagram which is then used as a basis for creating a UML profile. The profile is augmented with constraints to aid system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. The profile is then applied to a domain model of a safety-critical application aiding system suppliers in clearly demonstrating how the development artifacts of their system fulfill the compliance requirements of a standard. Constraints can be automatically checked to ensure full compliance. The approach is illustrated by presenting an excerpt of a case study to show how the concepts in the domain of sub-sea production control systems can be related to the evidence requirements in the IEC61508 standard. The IEC61508 standard was chosen for illustration as it is a generic standard that applies to multiple domains and a successful application of the approach to it is a good indication of the usefulness of the work.

Paper IV: Planning for Safety Evidence Collection: A Tool-Supported Approach Based on Modeling of Standards Compliance Information.
Davide Falessi ,Mehrdad Sabetzadeh, Lionel Briand, Emanuele Turella, Thierry Coq, Rajwinder Panesar-Walawege.
IEEE Software, 2012.

This paper addresses the goal of planning for certification. Safety-critical software-dependent systems such as those found in the avionics, automotive, maritime, and energy domains often need to be certified based on one or more safety standards. An important prerequisite for demonstrating compliance to software safety standards such as IEC61508 is the collection of safety evidence. Without an upfront agreement between the system supplier and

the certifier about the evidence that needs to be collected, there will invariably be important omissions, which will need to be remedied after the fact and at significant costs. In this paper, a flexible approach and a supporting tool are presented for assisting suppliers and certifiers in developing an agreement about the evidence necessary to demonstrate compliance to a safety standard. The approach is model-based; specifically, the safety standard of interest is expressed via an information model. The supporting tool, which is available online, takes this information model as input and helps system suppliers and the certifiers in reaching a documented and consistent agreement about the safety evidence that needs to be collected. Though the tool has not been formally evaluated, it is *not* a major (and disruptive) break from current practice, and provides a more effective way to do what is being done already through the manual completion of a plethora of different checklists and spreadsheets.

Paper V: CRESCO: Construction of Evidence Repositories for Managing Standards Compliance.
Rajwinder Kaur Panesar-Walawege, Torbjørn Skyberg Knutsen, Mehrdad Sabetzadeh, Lionel Briand.
Tool Demonstration paper at the 30th ACM International Conference on Conceptual Modeling (ER), 2011.

This paper addresses the goal of managing safety evidence electronically in an alternative method to that presented in paper III. A tool, CRESCO, is described for the **C**onstruction of Evidence **RE**positories for Managing **S**tandards **CO**mpliance. CRESCO draws on Model Driven Engineering (MDE) technologies to generate a database repository schema from the evidence requirements of a given standard, expressed as a UML class diagram. CRESCO in addition generates a web-based user interface for building and manipulating evidence repositories based on the schema. CRESCO is targeted primarily at addressing the tool infrastructure needs for supporting the collection and management of safety evidence data. A systematic treatment of evidence information is a key prerequisite for demonstration of compliance to safety standards, such as IEC61508, during the safety certification process. The goal was to show feasibility via a coherent combination of existing open-source technologies. While the current tool provides a flexible infrastructure for managing compliance evidence, further work is required to turn it into a tool that can be deployed in a production environment. In particular, consideration is being given to adding more sophisticated query facilities such that complex queries can be posed as well as professional reporting facilities in order to extract data from the database to create reports that can be directly given to the certification body.

Paper VI: Using Model-Driven Engineering for Managing Safety Evidence: Challenges, Vision and Experience.
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand.
1st International Workshop on Software Certification at the 22th IEEE International Symposium on Software Reliability Engineering (ISSRE), 2011.

This paper is a vision paper that presents the overall vision for addressing the research goals. Before a safety-critical system can be put into operation it must under go the process of certification. This paper discusses the challenges that are faced by system suppliers and

certifier when having to certify systems to safety standards. These challenges are based on our experience in working in and with industry. System supplier are required to prepare for certification based on the relevant industry standards that are textually expressed and are subjectively interpreted. The suppliers run the risk of not collecting the requisite information during the development of the system and having to do so after the fact, leading to large cost overruns and delays in deployment of systems. On the other hand, certifiers may receive a large collection of documents from the supplier with the hope that the certifier will find the required safety information (based on *their* interpretation of the standard). This results in the certifier having to invest a significant amount of time and effort sifting through the provided documents, and in many cases not finding what they were looking for. What is required is a structured and systematic procedure for certification where both parties can proceed in a timely manner, being aware of what information to collect and how to navigate easily through it.

This paper states the position taken in this thesis on how safety evidence should be characterized and managed. Specifically, the application of Model-Driven Engineering as an enabler for performing the various tasks related to safety evidence management is proposed. The work carried out on the specification of safety evidence requirements, upfront planning of evidence collection activities, tailoring of evidence information to domain-specific needs, and storage of evidence information is outlined. Models are proposed as the means to tackle the identified issues: they can be used to clarify the expectations of standards and present opportunities for automation of the certification process. To this end, an overview of the work carried out to show the potential of using model-driven engineering techniques for safety certification is presented.

Paper VII: Supporting the Verification of Compliance to Safety Standards via Model-Driven Engineering: Approach, Tool-Support and Empirical Validation.
Rajwinder Kaur Panesar-Walawege, Mehrdad Sabetzadeh, Lionel Briand.
Submitted to the Journal of Information and Software Technology (IST),2012

This paper is an extension of paper III, it provides a description of the approach for aligning standards to organization practices, the required tool support and substantial new empirical results demonstrating the feasibility and usefulness of our approach. A novel technique that guides system designers in establishing a sound relationship between the domain model for a safety-critical application and the evidence model for a certification standard is proposed. The approach makes use of UML profiles and builds upon mature MDE technologies and tailors them for specifying and automatically checking the constraints that must hold for compliance with safety standards. More precisely, a profile is developed based on the conceptual model of a given standard. The profile is then augmented with verifiable constraints that help system suppliers to systematically relate the concepts in the standard to the concepts in the application domain. The resulting relationship provides a clear route for the supplier to demonstrate how their development artifacts can be used for showing compliance to the standard. The contributions in this paper are: (1) A general approach that uses MDE techniques to aid preparation for certification; (2) the adaptation of general modelling tools to manage evidence for certification; and (3) the application of the approach in the context of sub-sea production control systems. The approach uses general MDE techniques in a novel way and can be

adapted to other standards.

The results of the case study presented show that the approach provided systematic guidance in relating the concepts in the domain of sub-sea production control systems to those of the IEC61508 standard. It allows the suppliers to work in a modular fashion on the different parts of the system whilst still having an overall picture of the entire system. The domain experts that were surveyed indicated finding benefit in adopting the approach in their work. The IEC61508 standard was chosen as the standard for the pilot case study as it is a generic standard that applies to multiple domains and a successful application of the approach to it is a good indication of the usefulness of the work.

6 Future Directions

The work in this thesis explored several directions for helping both suppliers and certification bodies with the process of certification.

For the main contribution in this thesis - a model-driven approach for preparing for certification, there are a number of issues that can be further explored in the future.

The first step is the creation of the conceptual model of a certification standard. The model of the IEC61508 standard that was created for the case study was validated by experts in a certification body that uses that specific standard. The experts provided feedback which led to the revision of the model. The revised model was then presented to a group of twenty-eight certification experts in an industry workshop. During this workshop, the modelling notation was explained and the model itself fully presented. A question and answer session was held. This session resulted in no further changes to the model. Additionally, the author went through an exercise of checking all the requirements in the standard and checking that each concept in the model originated in a requirement from the standard. This ensures that there were no extraneous concepts or relationships in the conceptual model that did not originate in the standard. The focus of all this work was to create a specific and explicit interpretation of the standard that can be used to aid certification. We did not focus on investigating ways of modelling the alternative interpretations that can results from the subjective interpretation of the standard. This is work that needs to be further examined along with a study into whether conceptual models can be used as a sole means for presenting standards. Would this type of representation make it easy to identify changes in standards when they are updated, thus providing a means of easily disseminating the changes.

When a product is certified, the version of the standard being used should be agreed upon upfront and stated. This helps in ensuring that should a new version of the standard be released in the middle of the product development, then the product is not impacted. Having said that, when the standard does evolve, the conceptual model and hence the profile of the standard is impacted - the profile will need to be updated to be in alignment with the new requirements of the standard. This impacts any previously elaborated domain model since the domain model will need to be elaborated again using the new profile. It is possible to use a previously elaborated domain model and remove the old profile from the model. This will leave all the elements but no stereotypes. Then the elaboration part can be repeated, but new elements are only added as necessary - mostly it will be re-stereotyping of existing elements. This could lead to finding elements that are no longer necessary and be a little more time conserving. However, it would be interesting to devise a means of detecting the changes and automatically analysing a previously elaborated model to detect the differences that have resulted due the the changes in the profile. I suppose this is a question of comparing two profiles and detecting the differences and highlighting these differences on the elaborated domain model - this would improve the re-usability aspect of elaborated domain models.

The domain model of the system can be quite large. UML provides the facility to create hierarchical models and break down even their visual representations into manageable sizes, it is necessary to decompose the models into usable chunks. In future work it would be useful to check whether some of the models could be elaborated automatically e.g., where an element is stereotyped as a software block and all the required relationships to certification elements are automatically generated and added to the model. The models could also be analysed to

show all certification elements that are still missing.

The model-driven approach presented in this thesis focusses on which certification artifacts are to be created and not on the contents of the artifacts themselves - even the certification standards do not go into detail about the contents of the artifacts. It is the work of the certification body to check that the contents of the artifacts generated are sufficient. This thesis focuses on the first step in the process of getting a system certified - on which artifacts are required. The next step is to ensure the appropriate contents of those artifacts. One direction to pursue is to highlight for each artifact, which specific techniques need to be employed in the creation of the content of that artifact. Usually the techniques are dependent on the safety integrity level of the component to which the artifact refers to. Thus based on the SIL level, the techniques could be recommended. This would be a first step in helping with the appropriate content of the certification artifacts, further work also needs to be done to help the certification body in the evaluation of collated evidence.

Safety critical-systems are sometimes certified to multiple standards and it should be explored whether the approach itself can be extended for the certification of a system to multiple inter-related standards.

Finally, as a minor technological addition, a report generator needs to be added to the current tool infrastructure in order to present the models in the form of reports for upper management. This would allow the provision of relevant information for a number of different actors in an organization.

In terms of the work carried out on systematizing the specification of the relationship between generic and derived standards, the work needs to be validated with a comprehensive case study. The work has been done for one specific derived standard but sometimes a derived standard draws from multiple standards and this aspect needs to be explored further.

The creation of the automated repository for certification data was a preliminary investigation into this direction. The generated repository was checked with a small data set but it also needs a comprehensive case study. In this work the constructed repository was developed based on the conceptual model of a certification standard, further work can be done to explore whether such a repository can also be populated with data from models that have been created based on the conceptual model. The repository can then be a means of linking this data to other tools that cannot access the data in the models directly.

The work on using conceptual models for creating a tool for planning for evidence collection can be further extended by combining it with the work done on automatically generating evidence repositories for compliance evidence from conceptual models. The direction to explore is whether the data in a generated repository can be automatically checked for compliance with the agreement created from the planning tool.

7 Conclusion

This thesis has presented solutions to a number of goals that were set forth in the domain of certification of safety-critical systems based on industry standards. The work is grounded on creating conceptual models of certification standards and using these models in a number of different ways to aid certification.

The first application is for explicitly showing the relationship between a generic standard and a domain specific one. A complete approach is presented on how this can be accomplished. The case shown is the relationship between the generic IEC61508 standard and the OLF070 standard for the petroleum industry.

Further two tools are presented - one that automatically generates an evidence repository from a conceptual model of a standard and another that uses this model to created a questionnaire that the supplier and certifier can use to come to an agreement about which artifacts will be created in readiness for certification.

Finally, an approach for showing compliance to a safety standard is presented. This approach establishes a sound relationship between a domain model of a safety-critical application and the evidence model of a certification standard. The relevant standard is captured as a conceptual model using a UML class diagram which is then used to create a UML profile. The profile is augmented with constraints to aid system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. The profile is then applied to a domain model of a safety-critical application, aiding system suppliers in clearly demonstrating how the development artifacts of their system fulfil the compliance requirements of a standard. Constraints are automatically checked to ensure full compliance. A complete case study for the feasibility of this work is given showing that the approach provided systematic guidance in relating the concepts in the domain of sub-sea production control systems to those of the IEC61508 standard. It allowed the suppliers to work in a modular fashion on the different parts of the system whilst still having an overall picture of the entire system. The domain experts surveyed indicated finding benefit in adopting the approach in their work. The successful use of IEC61508 as the standard for the case study is a good indication of the usefulness of the work as it is a generic standard that applies to multiple domains.

Bibliography

[1] *OMG Object Constraint Language*.

[2] *OMG Object Management Group*. http://www.omg.org/.

[3] *Meta Object Facility (MOF) Core Specification*. http://www.omg.org/spec/MOF/2.0/, August 2006.

[4] Y. BAI AND Q. BAI, *Subsea Engineering Handbook*, Elsevier, 2010.

[5] B. R. BRYANT, J. GRAY, AND M. MERNIK, *Domain-specific software engineering*, in Proceedings of the FSE/SDP workshop on Future of software engineering research, FoSER '10, ACM, 2010, pp. 65–68.

[6] B. R. BRYANT, J. GRAY, M. MERNIK, P. J. CLARKE, R. FRANCE, AND G. KARSAI, *Challenges and directions in formalizing the semantics of modeling languages*, Computer Science and Information Systems, 8 (2011), pp. 225–253.

[7] F. CHAUVEL AND J.-M. JÉZÉQUEL, *Code generation from uml models with semantic variation points*, in Proceedings of the 8th international conference on Model Driven Engineering Languages and Systems, MoDELS'05, Springer-Verlag, 2005, pp. 54–68.

[8] T. COCKRAM AND B. LOCKWOOD, *Electronic safety cases: Challenges and opportunities*, in Current Issues in Safety-Critical Systems, in proceedings of Safety Critical Systems Symposium, Springer, 2003.

[9] W. R. DUNN, *Practical Design of Safety-Critical Computer Systems*, Reliability Press, 2002.

[10] *EMF Eclipse Modeling Framework*. http://www.eclipse.org/modeling/emf/.

[11] R. FELDT, R. TORKAR, E. AHMAD, AND B. RAZA, *Challenges with software verification and validation activities in the space industry*, in ICST'10, 2010, pp. 225–234.

[12] D. GAÅĄEVIC, D. DJURIC, AND V. DEVEDÅ¿IC, *Model Driven Engineering and Ontology Development*, Springer, 2009.

[13] *IBM Rational Software Architect*. http://www.ibm.com/developerworks/rational/products/rsa/.

[14] *EN50128 - Railway Applications - software for railway control and protection systems*, 1999.

[15] *Functional safety - safety instrumented systems for the process industry sector (IEC 61511)*., 2003.

[16] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[17] *Road vehicles – functional safety*, 2009. ISO draft standard.

[18] A. G. KLEPPE, *A language description is more than a metamodel*, in Proceedings of the Fourth International Workshop on Software Language Engineering, Springer-Verlag, 2007.

[19] L. LAVAGNO, G. MARTIN, AND B. SELIC, eds., *UML for real: design of embedded real-time systems*, Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[20] O. NORDLAND, *A critical look at the cenelec railway application standards*. `http://home.c2i.net/odd_nordland/~SINTEF/tekster/A_critical_look_at_rail_standards.htm`, 2003.

[21] B. OATES, *Researching information systems and computing*, SAGE, 2006.

[22] *UML 2.0 Superstructure Specification*, August 2005.

[23] *DO-178B: Software considerations in airborne systems and equipment certification*, 1982.

[24] F. REDMILL, *Installing IEC 61508 and supporting its users – nine necessities.*, in 5th Australian Workshop on Safety Critical Systems and Software, 2000.

[25] P. RUNESON AND M. HÖST, *Guidelines for conducting and reporting case study research in software engineering*, Empirical Softw. Engg., 14 (2009), pp. 131–164.

[26] N. SANNIER, B. BAUDRY, AND T. NGUYEN, *Formalizing standards and regulations variability in longlife projects. a challenge for model-driven engineering.*, in MoDRE workshop, 2011, pp. 225–234.

[27] *Application of IEC61508 and IEC61511 in the Norwegian Petroleum Industry*, 2004.

Paper I:
Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard

Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard

Rajwinder Kaur Panesar-Walawege[1,2], Mehrdad Sabetzadeh[1], Lionel Briand[1,2], Thierry Coq[3]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

[3] Det Norske Veritas,
Paris France

Abstract:

Increasingly, licensing and safety regulatory bodies require the suppliers of software-intensive, safety-critical systems to provide an explicit software safety case – a structured set of arguments based on objective evidence to demonstrate that the software elements of a system are acceptably safe. Existing research on safety cases has mainly focused on how to build the arguments in a safety case based on available evidence; but little has been done to precisely characterize what this evidence should be. As a result, system suppliers are left with practically no guidance on what evidence to collect during software development. This has led to the suppliers having to recover the relevant evidence after the fact – an extremely costly and sometimes impractical task. Although standards such as the IEC 61508 – which is widely viewed as the best available generic standard for managing functional safety in software – provide some guidance for the collection of relevant safety and certification information, this guidance is mostly textual, not expressed in a precise and structured form, and is not easy to specialize to context-specific needs. To address these issues, we present a conceptual model to characterize the evidence for arguing about software safety. Our model captures both the information requirements for demonstrating compliance with IEC 61508 and the traceability links necessary to create a seamless chain of evidence. We further describe how our generic model can be specialized according to the needs of a particular context, and discuss some important ways in which our model can facilitate software certification.

1 Introduction

Safety-critical systems such as those found in the avionics, automotive, maritime, and energy domains are often required to undergo a safety certification process. The goal of certification is to provide an assurance recognized by society (and in some cases by law) that a system is deemed safe by the certification body.

The justification for safe operation of a system is usually presented in what is known as a safety case [3, 13, 14, 16, 21]. Kelly [14] describes a safety case as being composed of three principal parts: safety objectives, arguments, and evidence. Demonstrating the satisfaction of the objectives involves gathering systematic evidence during development and constructing well-reasoned arguments that relate the evidence to the objectives.

With the growing use and complexity of software in safety-critical systems, licensing and safety regulatory bodies increasingly require system suppliers to provide an explicit software safety case. A software safety case is a part of an overall safety case, which provides assurance that the software elements of a system are sound, and that these elements are used correctly within the overall system.

While the argumentation aspects of software safety cases (and generally, safety cases) have been studied for a long time [18]; little has been done to precisely characterize the evidence that underlies software safety arguments. As a result, suppliers of safety-critical software have been left without proper guidance on what evidence to collect during development. This has led to the suppliers having to recover the relevant evidence after the fact, which can be extremely costly or even impractical. In addition, the quality of the overall safety case is bound by the quality of the weakest link. Hence, current practices for managing software safety evidence can severely limit the effectiveness of safety cases.

Although standards such as IEC 61508 [12] – which is widely viewed as the best available generic standard for management of functional safety in software – provide some guidance for collecting safety and certification information, this guidance is mostly textual, not expressed in a precise and structured form, and is not easy to specialize to context-specific needs.

The goal of this paper is to address the above issues by providing a conceptual model that characterizes the evidence necessary for arguing about software safety. Our model captures both the information requirements for demonstrating compliance with IEC 61508, and the traceability links necessary to create a seamless continuum of evidence information, called the *chain of evidence* [13].

In real-life projects, multiple rules, regulations and standards apply; therefore, our conceptual model needs to be further specialized according to the safety needs of the application domain (e.g., national and international laws, and class society regulations in the maritime domain [8]), the development process, and the technologies used to express requirements and design decisions (e.g., SysML [10]). A specialized version of the conceptual model can in turn be used for constructing an evidence repository. Such a repository can be utilized for automating various development and analysis tasks associated with safety-critical software, including safety report generation, checking of various compliance rules, and impact analysis.

The remainder of this paper is structured as follows: In Section 2, we give a brief introduction to the IEC 61508 standard. We provide a detailed exposition of our conceptual model in Section 3; and in Section 4, we exemplify some key aspects of the model. In Section 5, we explain how the model can be specialized according to the needs of a particular context.

In Section 6, we describe some important applications of the model in software certification. Section 7 provides initial validation of the usefulness of our model. Section 8 compares our work to related research; and Section 9 concludes the paper with a summary and directions for future work.

2 Background

This section provides background information on the IEC 61508 standard (version published in 1998). The standard is concerned with improving the development of safety-related electrical/electronic/programmable electronic systems (E/E/PES) whose failure could result in harm to people, equipment, and/or the environment. IEC 61508 is a generic standard and can either be used directly or for the creation of domain-specific standards in industries that require an equivalent level of safety.

The standard applies to both low-demand and continuous mode systems. In a low-demand system, the frequency of demands for operation is low (the standard specifies a precise range). An example of a low-demand system is a fire & gas protection system, which alerts personnel if a fire or gas leakage is detected and initiates protective actions either automatically or through manual intervention. A continuous (or high-demand) mode system is one with a high frequency of demands for operation. An example would be the dynamic positioning system that continuously controls a vessel's movement when the vessel is near a port or rig.

The goal of the standard is to ensure that safety-related E/E/PES systems operate correctly in response to their inputs. This is referred to as *functional safety*. Functional safety is not all there is to safety. For example, the activation of an alarm in response to a fire breakout is a functional safety measure, whereas the use of fire resistant walls to control the spread of fire is not, although the latter measure protects against the same hazard. IEC 61508 deals only with functional safety. A function that a control system performs to ensure that the system remains in a safe state is referred to as a safety function. Each safety function specifies what safety objective is to be achieved (safety function requirement) and the level of integrity with which the safety function is implemented (safety integrity level).

To systematically deal with the activities necessary to achieve the required level of safety, the standard adopts an overall safety lifecycle. The lifecycle starts with establishing the concept and overall scope of a system, and then conducting a hazard and risk analysis to determine the hazards that can occur and the risks that they pose. Together, these activities determine what has to be done to avoid the hazardous situations (derivation of safety requirements) and the level to which safety has to be provided (derivation of safety integrity levels).

In the next step, the safety requirements are allocated to the various designated E/E/PE safety-related systems, other technology safety-related systems, and external risk reduction facilities (only the E/E/PE allocations are within the scope of the standard). Once the allocations are made, the realization phase begins for both the hardware and software aspects of the E/E/PE safety-related systems. In tandem, planning begins for the installation and commissioning, operation and maintenance, and the final overall safety validation of the system. During the realization phases, the standard calls for a number of overarching verification, management, and assessment activities. The life cycle further takes into account the eventual, safe, decommissioning or retrofit of the system.

In this paper, we deal with the activities that take place during the realization of the software part of a programmable electronic safety-related system. The standard requires an explicit *software* safety lifecycle, shown in Figure 1, for the development of a PES.

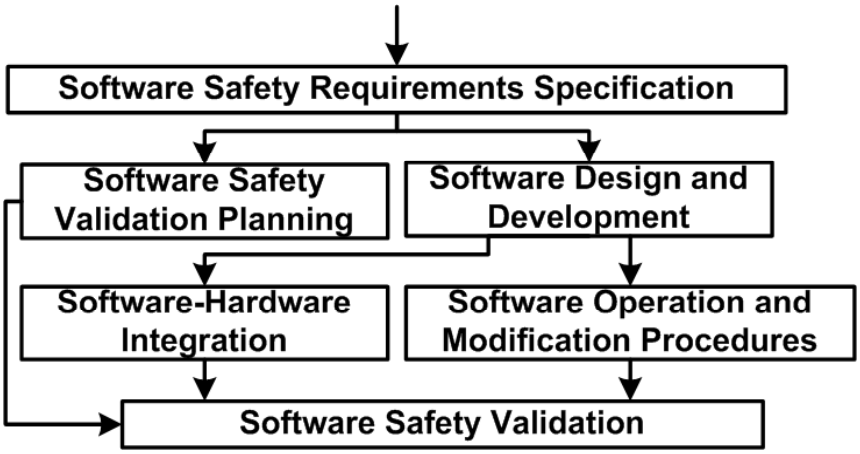


Figure 1: IEC 61508 Software Safety Lifecycle

The lifecycle for the realization of the hardware in the E/E/PES is similar except that it applies to the hardware. It is important to realize that the hardware and software development lifecycles are happening in parallel and certain hardware architectural assumptions will have to be in place before the relevant software lifecycle can be started.

The software has to be implemented such that it fulfills the safety requirements allocated to it. In order to be able to show this during software safety validation and assessment, it is crucial to maintain traceability between the software safety requirements, and the decisions taken during design, and the actual implementation in code. This is a complex task and needs to be performed whilst the system is being developed, not once the development has finished. Providing an accurate description of the safety information that needs to be preserved during software development is the main motivation behind our work in this paper.

The software safety lifecycle in Figure 1, together with the overall lifecycle activities (verification, management and assessment of safety) specialized to software, form the basis of the conceptual model in Section 3.

3 Conceptual Model

Figure 2 formalizes our conceptual model as a UML class diagram. The concepts in the model are only succinctly and intuitively defined here and precise definitions are provided in a technical report [4]. To manage the apparent complexity of the model, the concepts have been divided into ten packages. We describe these packages next. Note that this conceptual model is meant to define, in a precise way, information requirements to demonstrate both compliance with the standard and, perhaps more importantly, ensure the safety chain of evidence is collected.

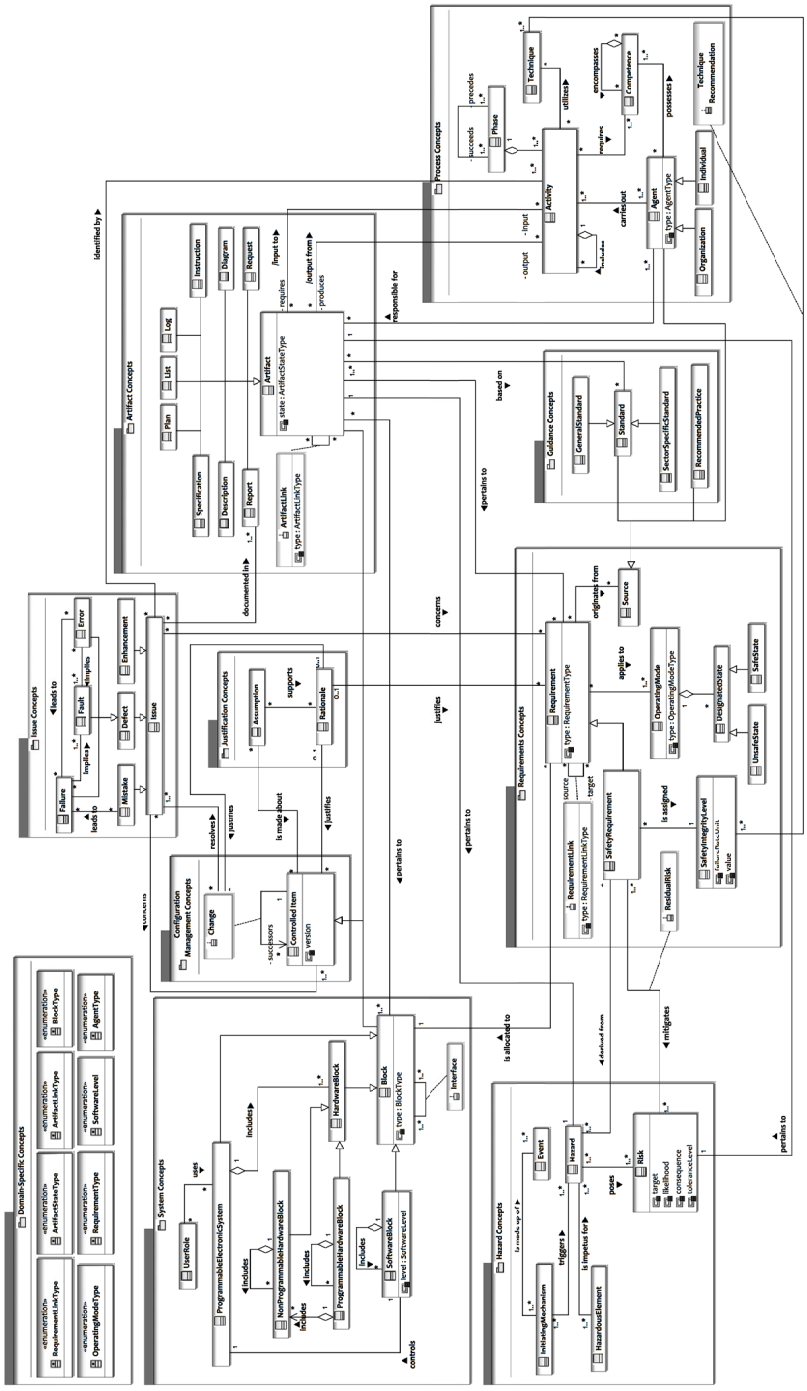


Figure 2: Core Concepts and Relationships

3.1 System Concepts

The System Concepts package describes the basic elements needed to conceptualize safety-related control systems that involve both hardware and software. A Programmable Electronic System (PES) is a block made up of one or more hardware blocks and controlled by a number of software blocks. A hardware block may represent a mechanical, electrical or electronic entity, both programmable and non-programmable. Both hardware and software blocks can be hierarchically decomposed into lower-level blocks. For software, the typical decomposition levels are: module, component, subsystem, and system. The links between blocks and the corresponding development artifacts (see Section 3.5) are captured through the association between the Block and Artifact concepts.

Interactions between the blocks are expressed as interfaces. Making the interfaces explicit is necessary to minimize mismatches and deadlocks during integration. For arguing software safety at the level of an individual PES, the interfaces of interest are those that have a software block at least at one end (i.e., no hardware-to-hardware interfaces). For integration of system-of-systems, interfaces between PESs are crucial as well.

Interactions between a PES and the human elements are modeled through user roles. Safety issues can arise due to misuse or unauthorized access to a system. Mitigating these issues requires an accurate description of how different groups of users can interact with the PES.

Each block is traceable to the requirements allocated to it. At the PES level, the allocations are made during the safety requirements allocation step of the IEC 61508 overall safety lifecycle. The PES-level (safety) requirements are used to derive requirements for the software and hardware blocks. We discuss requirements in Section 3.3. Blocks can evolve over time and are thus versioned and placed under configuration management. Configuration management is addressed in Section 3.7.

3.2 Hazard Concepts

The Hazard Concepts package captures the hazards and the risks they pose, which then constitute grounds for safety requirements and safety integrity levels. A hazard is any real or potential condition that can cause injury, illness, or death to personnel; damage to or loss of a system, equipment or property; or damage to the environment.

The potential for a hazard to occur exists whenever the system has some hazardous element in it – this is the basic hazardous resource creating the impetus for the hazard. An example could be a hazardous energy source such as explosives. The hazardous element in itself is not sufficient to trigger a hazard. The trigger is captured using the concept of an initiating mechanism. An initiating mechanism is a sequence of events that leads to the actualization of a hazard. Hazards are the basis for deriving safety requirements.

Each hazard is analyzed to assess the risks it poses, using risk assessment techniques. In essence, a risk is the combination of the probability of occurrence of a particular harm and the severity of that harm to a person or object, usually referred to as the target.

The probability of occurrence is referred to as the likelihood and is sometimes qualitatively divided into: frequent, probable, occasional, remote, improbable and incredible. The level of harm caused is referred to as the consequence and can be qualitatively rated as catastrophic, critical, marginal or negligible. Together, these are used to give a tolerance level to a risk. The level of tolerance of a risk is then used to derive a safety integrity level. The results of

hazard and risk analysis are presented as a Description. Hazards and risks can be referred to in various other development artifacts such as requirements specifications.

3.3 Requirements Concepts

The concepts necessary to describe the requirements for creating, operating, maintaining and decommissioning a system are included in the Requirements Concepts package. Traceability from requirements to the corresponding PES, system blocks, hazards and artifacts forms an important part of the chain of evidence.

A requirement is a statement identifying a capability, characteristic, or quality factor of a system in order for it to have value and utility to a user. Requirements are one of the central concepts of system development and are thus naturally connected to concepts in many other packages. A requirement is typically concerned with some particular aspect of the system (functionality, usability, performance etc.). This information is captured in the "type" of the requirements. Each requirement is linked to the block(s) that must realize it. A rationale item might be affixed to a requirement to justify why that requirement exists. If an issue is raised about a requirement at some stage of development, the issue is recorded and linked to the requirement as well. The source of a requirement may be a person, organization, standard or recommended practice. A requirement may apply to certain operating modes of the system such as normal operation, maintenance, shut down, and emergency. Each operating mode may have a set of designated states, which would render the system safe or unsafe. For example, it might be unsafe to run a boiler engine during maintenance.

A particular class of requirements is that which concerns safety. Safety requirements are used to ensure that the system carries out its functions in an acceptably safe manner. These requirements are derived from hazards, and are intended to mitigate the risks posed by these hazards. Each safety requirement is assigned a safety integrity level based on the likelihood and consequences of the risks it mitigates.

Safety integrity is defined as the probability of the system to successfully perform a required safety function. Usually, the dual notion of probability of failure (instead of probability of success) is used. The failure rate unit can be "failure per hour" for high demand or continuous operation and "failure on demand" for low demand operation. When a safety requirement only partially addresses a risk, the residual risk (i.e., the risk fraction remaining after the protective measures have been taken) is recorded.

A requirement may relate to other requirements in a number of ways. Example relationships include: when a lower-level requirement (e.g., module requirement) is derived from a higher-level requirements (e.g., system or component requirement), when a requirement contributes positively or negatively to the satisfaction of another requirement, and when a requirement conflicts with or overrides another requirement. In these cases, we need to maintain traceability between the involved requirements. This is done using a reflexive association for the Requirement concept.

A requirement can have various development artifacts associated with it. Particularly, a requirement is specified in some requirements specification, and referenced in many other artifacts such as design and architecture specifications, test plans, source code, and also other requirements specifications where related requirements are captured.

3.4 Process Concepts

Development of software for a PES follows a certain process. This is expressed using the Process Concepts package. Further refinements of the process concepts would have to be performed in specific contexts of applications, accounting for the specifics of the process in place.

The notion of activity is the central concept in this package, representing a unit of behavior with specific input and output. An activity can be further decomposed into sub-activities. A (lifecycle) phase is made up of a set of activities that are carried out during the lifetime of a system, starting from system inception to decommissioning. To be able to accommodate iterative development processes, we do not restrict activity types to particular development phases. Restrictions will be expressed externally where necessary, for example using OCL constraints [1].

Each activity utilizes certain techniques to arrive at its desired output, given its input. The selection of techniques is intimately related to the safety integrity level that needs to be achieved. For example, if the activity in question concerns software verification, constructing formal proofs of correctness is usually unnecessary for low integrity levels, whereas, formal proofs are highly recommended (and often necessary) for the highest integrity levels. Specific technique recommendations (e.g., recommended, not recommended, highly recommended, mandatory) are made based on the overall standard guidelines, and the requirements of the certification bodies in charge of assessing functional safety.

Each activity requires certain kind of competence by the agents performing it. The agent itself can be either an individual person or an organization. In either case, the agent is identified by the type of role it plays, for example the agent may be the supplier of a system or the operator. Agents can be made responsible for certain development artifacts.

3.5 Artifact Concepts

The Artifact Concepts package characterizes the inputs and outputs of the development activities. The main concept here is Artifact, which describes the tangible by-products produced during development. IEC 61508 provides a high-level classification of the different types of development artifacts: a *specification* (e.g. requirements specification); a *description* (e.g. description of planned activities); a *diagram* (e.g. architecture diagram); an *instruction* (e.g., operator instructions); a *list* (e.g., code list, signal list); a *log* (e.g., maintenance log); a *plan* (e.g., maintenance plan); a *report* (e.g., a test or inspection report); and a *request* (e.g., a change request).

An artifact might be built based on a standard, e.g., source code may follow a certain coding standard. Each artifact can pertain to requirements, blocks, hazards, and risks, as discussed in earlier sections. An artifact can be linked to other artifacts as well. For example, a design document may realize the requirements in the requirements specification, or a report could be the result of carrying out a plan. Issues that are identified during lifecycle activities are documented in reports. Like system blocks, artifacts can evolve over time and are therefore versioned and under configuration management.

IEC 61508 prescribes specific input and output artifacts for all the activities in the overall lifecycle. As an example, we have shown in Figure 3 the input and output artifacts for the Software Module Testing activity, whose goal is to verify that each software module performs

its intended function and does not perform unintended functions. In the technical report version of this paper [4], we provide conceptualizations similar to that in Figure 3 for all the software lifecycle activities.

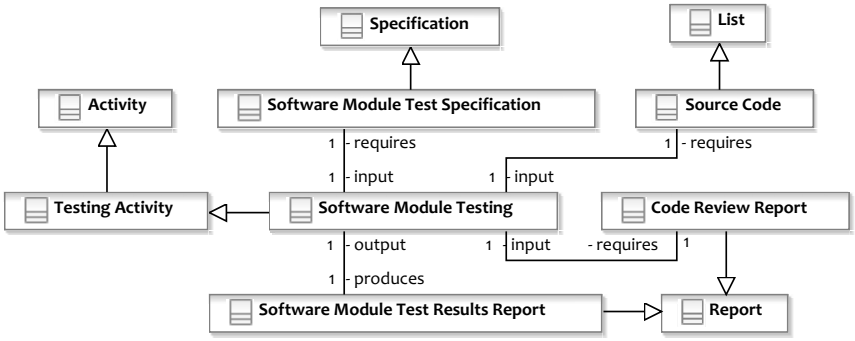


Figure 3: Software Module Testing Activity

Note that the links between the more specific subclasses of Artifact and these lifecycle activities (e.g., the link between Source Code and Software Module Testing in Figure 3) refine the high-level input and output links between Artifact and Activity in the conceptual model. Therefore, in Figure 2, the links between Activity and Artifact can be seen as derived (hence the '/' before the link names). Further, note that the various artifacts in the standard need to be specialized in any given context. For example, the Software Module Test Specification in Figure 3 could be defined as being composed of test cases that exercise certain blocks or requirements. Similarly, the notions of test stub, and test driver could be made explicit for testing. Deciding about how much structure to enforce on each artifact is one of the key aspects of specialization (see Section 5).

3.6 Issue Concepts

The concepts enabling the description of issues are modeled in the Issue Concepts package. Issue is the broad term we use to denote a point in question or a situation that needs to be settled in regards to a controlled item or a requirement (controlled items are discussed in 3.7). Issues may represent defects, human mistakes, or enhancements and can be a result of activities concerned with Verification & Validation (e.g. testing and inspection) and safety assessment. In addition, enhancement may be proposed at different stages of development as a result of activities such as requirements engineering and design, or in response to the findings of V&V and safety assessment. Defects can be further refined into errors, failures and faults. An error is a discrepancy between the actual and the ideal output. IEC 61508 distinguishes system errors from human errors, referred to as mistakes. Mistakes denote unintended results due to human action or inaction. A failure is defined as the inability of a unit to perform a required function, and a fault as the abnormal condition that causes a unit to fail (e.g., a software bug).

To illustrate these concepts, consider a boiler system. An error could be when the observed temperature is 80 degrees Celsius while the water is boiling, i.e., when the expected value is

100. If there is a safety requirement stating that the boiler should activate the pressure-release valve in case of over-heating (i.e., when the temperature has reached 100), then the error would lead to a failure, because the safety function would not be delivered. An error does not necessarily lead to a failure. In our example, if the actual temperature was 80 and the observed one was 60, there would still be an error but no failure. Failures and errors might imply faults. In our example, the fault could be a damaged sensor or the boiler's control unit incorrectly interpreting the temperature sensor output.

Mistakes made by an operator of the system can lead to failures. For example, if the safety function requires manual intervention and the operator fails to notice the alarm indicating an over-heating boiler, he would not engage the safety function. Mistakes may lead to changes to the operating procedures, or even the system. For example, the operating procedure may be changed to ensure that at least one operator is monitoring the control panel at all times; or the system's user interface may be revised to reduce the possibility of alarms going unnoticed.

The decision made about an issue (whether it is valid, and if so, how it has been resolved) is documented in a report. The resolution of an issue may induce change to some controlled items. Note that issues can be raised not only through the development stage, but also during operation, maintenance, decommissioning, etc.

3.7 Configuration Management Concepts

Valid issues need to be addressed through change. The concepts required for management of change and for ensuring that the safety requirements continue to be satisfied as the system evolves are captured in the Configuration Management Concepts package. Demonstration of accurate change management is necessary for compliance with IEC 61508. The central concept here is a controlled item, which is any item for which meaningful increments of change can be documented. In our model, blocks, artifacts and PESs are controlled items. Each controlled item may have some rationale to justify its existence, and assumptions to describe constraints or conditions about the item. Assumptions and rationale are further explained in Section 3.8. Changes to controlled items are made in response to issues, as discussed earlier, and can be justified by rationale.

3.8 Justification Concepts

System development entails various decisions which need to be justified by reasoning and based on assumptions about the domain and the artifacts. The basic concepts to enable justification are provided in the Justification Concepts package. There are two concepts here, assumption and rationale. An assumption is a premise that is not under the control of the system of interest, and is accepted as true. A rationale is the reason that explains the existence of a controlled item or a requirement in the system. The rationale may rely on some of the assumptions that have been made about the concerned block or artifact. An assumption about a PES as a whole will have overarching affects whereas assumptions regarding a particular block may affect how it is designed and implemented. In safety-critical systems, assumptions play a key role. In particular, most statements about the safety of a system are tied to the assumptions made about the environment where the system will function [13].

3.9 Guidance Concepts

Many aspects of development are influenced by guidance from external resources. For example, a sector-specific standard or a recommended practice may mandate certain requirements that must be fulfilled by the PES; or the implementation source code may be expected to be based on a certain coding standard. Such external resources are captured using the Guidance Concepts package. The guidance package describes the various sources of advice and recommendations used throughout development. A standard provides formal recommendations on engineering or technical criteria, methods, processes and practices and can be either general such as IEC 61508 or sector-specific such as ISO 17894 [2] that provides principles for the development and use of PESs in marine applications . The recommended practice on the other hand may be much more prescriptive and specific, providing sound practices and guidance for the achievement of a particular objective. Either may be used as a measure of compliance.

3.10 Domain-Specific Concepts

Finally, the Domain-Specific Concepts package contains enumeration types that can be customized by defining specific enumeration values for a given context. The concepts behind the enumerations have already been described in the other package descriptions. In Figure 4, we show examples of the kinds of values that can be used for each enumeration type.

4 Illustrating the Chain of Evidence

The conceptual model described in the previous section gives an overall view of the safety evidence pieces and the interconnections that need to be established between these pieces during the lifecycle of a safety-critical system.

Figure 5 shows a partial instantiation of the concepts in the model and their links. The hazard shown is the breakout of fire on an oil platform. The hazardous element involved is the combustible gas on the platform. The initiating mechanism leading to a fire breakout is the presence of a gas

leak and a spark in the vicinity of the leak. The hazard is identified during a hazard analysis activity and documented in a hazard log. For every hazard, a risk analysis activity is conducted and a report indicating the risks to mitigate is created. Two of the potential risks that such a fire can pose are damage to the platform and loss of life.

Based upon the hazard, safety requirements are derived and allocated to the various risk mitigation facilities. One such facility is the fire & gas protection system. The safety requirement allocated to this PES is that it must detect a fire breakout within two seconds of occurrence. A safety requirement for the software system is then derived for the software system that controls the PES, stating that the time from the actual detection of fire from the sensor until an alarm (visual and/or aural) is presented on the operator control panel is less than one second. This requirement is further partitioned between the control software and the heat sensor driver. The requirement allocated to the sensor driver is that it must keep the delay between two consecutive polls of the sensor to less than 200 milliseconds.

In this example, we can see the relationships between the different blocks, the requirements associated with each block, the derivation of lower-level requirements from higher-level

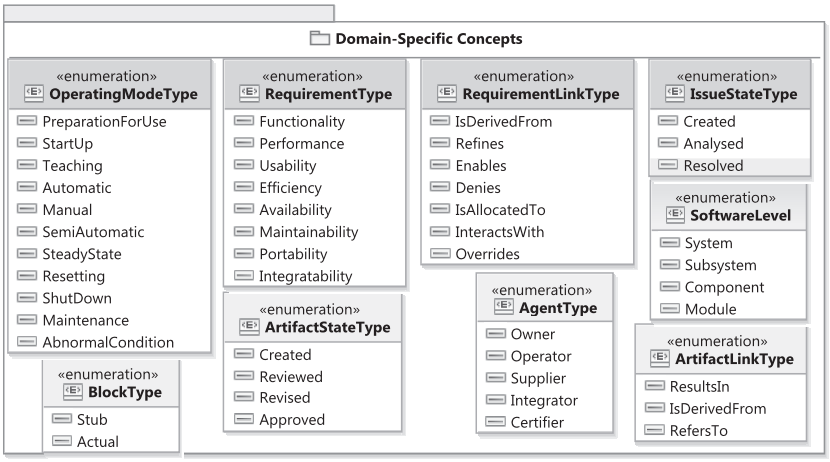


Figure 4: Example Values for Domain-Specific Enumerations

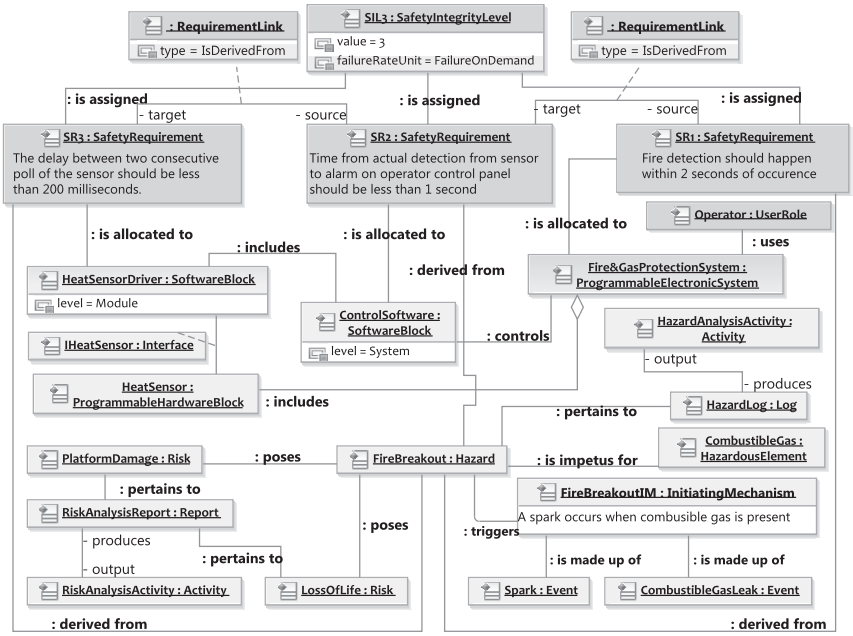


Figure 5: Example Evidence Information

requirements, the root hazard and associated risks, and the lifecycle activities. The example could have been expanded to show a variety of other activities (e.g., design and testing) and artifacts (e.g., design specifications, test specifications and test results). All this information needs to be accounted for when a software safety case is being developed.

5 Specialization of the Conceptual Model

IEC 61508 is a generic standard and can be implemented and augmented in a variety of ways depending on contextual factors, including the characteristics of a particular application domain, and the development process and technologies to be used. Specialization is an important prerequisite for developing a coherent, IEC 61508-compliant safety information model, which can guide data collection and support analysis in a particular development context. The generic conceptual model we developed in Section 3 provides an intuitive and technically rigorous basis for describing specializations. As an example, we show how to define a special type of the Diagram artifact (see Section 3.5), and use this specialized diagram for expressing Assumptions (see Section 3.8).

In a safety-critical system, it is important to state the assumptions (e.g., about the operating environment) in a way that permits systematic analysis. This helps ensure that we can assess the validity of requirements, specifications, and design decisions and to verify that there are no conflicts between the required system properties [13]. A powerful and flexible notation for formalizing assumptions is the Parametric Diagram in the SysML modeling language [10]. This type of diagram is used for representing constraints on a system's property values. In Figure 6, we have shown an example parametric diagram.

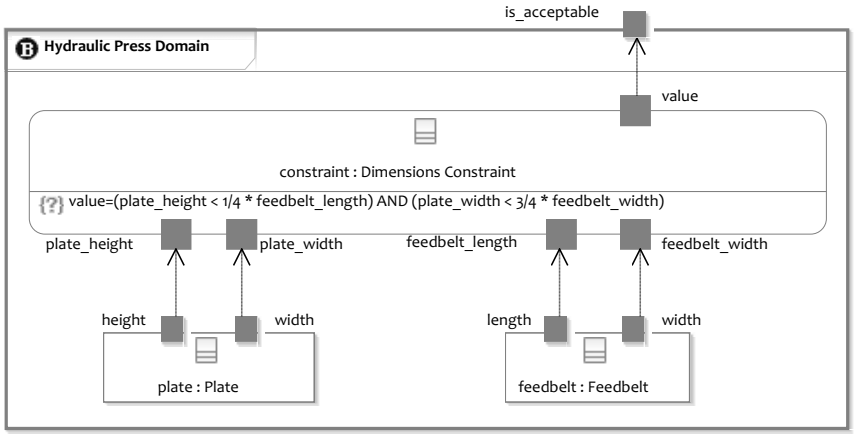


Figure 6: Parametric Diagram for an Assumption

The diagram describes a domain assumption about the physical dimensions of the plates that are fed to a hydraulic forging press. The assumption states that the height of a plate is no larger than ¼of the length of the feed belt that conveys the plate to the press, and that the width of a plate is not larger than ¾of the width of the feed belt. The former constraint is to

ensure that the plate is small enough to be picked up by the robot arm that places the plate on the press table, and the latter – to ensure that plates would not fall off the edges of the feed belt while in motion.

If we want to develop a specialized standard or recommended practice requiring that a parametric diagram should be constructed for every assumption, our conceptual model will be extended as follows: A Parametric Diagram is defined as a subclass of Diagram and an association is established between Assumption and Parametric Diagram. This is depicted in Figure 7.

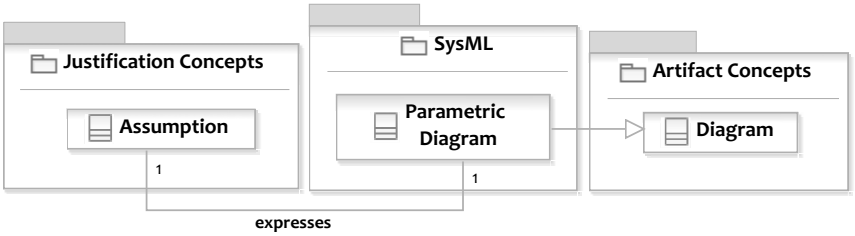


Figure 7: A Specialization of the Generic Model

In general, specialization refers to the extensions one makes to the conceptual model of Figure 2 in order to adapt it to a particular context. The extensions can be made by adding new classes (or subclasses), associations, attributes, and constraints. The example in Figure 7 already shows the addition of new (sub)classes and associations to the model. Below, we illustrate some simple extensions through new attributes and constraints. The model in Figure 2 is intentionally abstract, thus only providing the attributes that are fundamental to understanding the concepts. Any specialization of the model into an applicable, context-specific information model necessarily requires many new attributes to be defined. For example, most concepts need a universal identifier (uid), a name, and a description attribute. Constraints will be used frequently in the specializations of the model as well. For example, IEC 61508 highly recommends that module testing (see Figure 3) for safety integrity level 4 (SIL 4) should utilize probabilistic testing. If the certification body applying the standard wants to make this mandatory, it may choose to add the following OCL constraint to the model in Figure 2:

```
context SafetyIntegrityLevel
inv:
self.forAll(sil.value = 4 implies
  sil.SafetyRequirement.Block-> forAll(
    b.SoftwareModuleTestResultReport.output.
      Technique-> exists(t.name = "Probabilistic Testing"))
```

The above constraint states that a module testing activity associated with a block that has SIL 4 requirements must utilize the probabilistic testing technique (we have assumed that each technique is identified by a name attribute).

A full specialization of our conceptual model will involve numerous extensions like the ones illustrated above. Once a full adaptation of our model to a particular context is arrived at,

the resulting model can be used to drive data collection during the development process and to automate some of the most important and yet laborious tasks in the software certification process, as we discuss in the next section.

6 Applications

Having described our conceptual model and how it can be specialized, we now discuss some important ways in which our conceptual model or its specializations can facilitate software certification.

6.1 Aid to Understanding and Communicating IEC 61508

At the most basic level, the conceptual model we have developed helps improve understanding and communication of the IEC 61508 standard. Interpreting a standard like IEC 61508 is a daunting task for a software supplier. Even when attempted, the interpretation of a supplier is likely to be significantly different from that of the certifier. An issue we have noticed through our interactions with safety-critical software suppliers is that it is not always clear to them what documents and information, and at which level of detail, they are expected to provide in support of safety. Furthermore, it is frequently unclear what the scope of each document should be, and how these documents should be linked to hazards, requirements, activities, and so on. These issues typically lead to several unnecessary iterations to refine and complete certification documents as well as many misunderstandings that could have been avoided.

A concise but precise graphical representation of the core concepts in the standard such as the one we have developed here is a valuable and appealing aid for understanding and using the standard. In particular, the representation can be used by the certifiers to convey their expectations and to clarify the information requirements for demonstrating compliance.

6.2 A Contract between Suppliers and Certifiers

After our conceptual model has been specialized to a particular context, it can be viewed as a formal contract between the supplier and the certifier. Specifically, from the specialized conceptual model, the supplier can determine what evidence the certifier expects, and can accordingly plan its data collection strategy. In the absence of such a contract, the supplier will not know a priori what evidence needs to be collected. This often leads to the supplier failing to record important information during the development process, and having to incur significant costs to recover the information later on. Having such a contract is also advantageous to the certifier, because it permits them to assume that the safety evidence provided to them has certain content and structure.

Hence, the certifier can optimize its safety assessment procedures according to the content and structure mandated by the contract. For instance, the specialization example we gave in Section 5 would bind the supplier to use a SysML parametric diagram for expressing assumptions. Hence, the supplier would know exactly how to express and record assumptions during development and the certifier would know exactly what they can expect during assessment and possibly build supporting analysis tools to analyze the consistency and completeness of assumptions.

Finally, the existence of a formal contract for safety evidence means that certifiers and suppliers may define a common language for the electronic interchange of data, for example based on XML. This offers an opportunity for automation of some laborious certification tasks, as we are going to describe below in Sections 6.3 and 6.4.

6.3 Automatic Safety Report Generation

Our conceptual model provides an abstract structure for the organization of software safety information. Once tailored to a particular context through specialization, the resulting concrete structure can be used as the basis for a repository for storing development artifacts, process knowledge, hazard analysis data, safety audits, etc. This repository can be queried automatically for safety-relevant information, and the results then assembled into safety reports. For example, the links in Figure 5 can be traversed from the hazard to all the related elements, and a structured document can be generated to facilitate the verification of all the safety evidence related to the hazard. Modern model-driven technologies already enable the development of such infrastructure.

The main traceability requirement for generation of safety reports has to do with how software development artifacts (e.g., software requirements, architecture, design, and tests) are linked to the higher-level safety concepts such as hazards, environmental and domain assumptions, and overall safety requirements. Establishing this traceability is a key issue that one must consider when the conceptual model is being specialized to a given context and the specific artifacts to be used in that context are being defined.

We believe that using Model Driven Engineering (MDE) will facilitate the definition and exploitation of the traceability information that can be used for automatic software safety report generation. To illustrate this point, let us revisit the example we gave in Section 5. The parametric diagram in Figure 6 not only alleviates any potential ambiguities about the textual description of the assumption, but also yields precise traceability links between the assumption and the involved system blocks, namely Plate, Feedbelt, and Hydraulic Press Domain (a super block). Hence, the chain of evidence is always maintained in a precise manner and can thus be queried automatically.

In contrast, if text-based documents are used, traceability cannot be strictly enforced. Further, the semantics of any traceability links established in text would not be machine-interpretable. As a result, the information cannot be precisely queried, without the danger of following false links or failing to follow the correct ones.

6.4 Automation for Rule Checking

An interesting use of our conceptual model is to define consistency and completeness rules based on the concepts in the model (or a specialization thereof) and then check these rules against the information in an evidence repository (see Section 6.3 where we discussed such a repository). Here, we give two simple rules that can be articulated directly over the generic conceptual model of Figure 2:

- From the model, we see that each activity requires certain competence, and that each agent possesses certain competence. This coupled with the fact that competence itself

can be defined in terms of encompassing other competence, can be used to define a rule for checking that an agent responsible for carrying out a given activity has the necessary competence to do so.

- Another example would be checking that each safety requirement derived from a hazard has indeed been allocated to a PES. This helps ensure that all derived safety requirements have been dealt with appropriately by some system component.

To fully realize this notion of automated rule checking, we need to have in place a specialized conceptual model based on which all the rules of interest can be articulated. We further need some type of rule checking engine that allows both the definition of the rules in some language and the verification of the rules written in that language against the development information of a system. For example, MDE technologies such as the Eclipse Modeling Framework [20] and its associated OCL engine [1] readily provide such capabilities. This ability is useful to both the suppliers and the certifiers of safety-critical systems. From the perspective of the supplier, the rules can be used to ensure that the system has been built according to some industry-specific standard or recommended practice, or even to perform impact analyses whereby specific rules could be defined to predict the impact of changes based on dependency information. From the perspective of the certifier, the rules could be defined such that the supplier provides the data from the model to the certifiers, according to some predefined interchange format, and the certifiers have some proprietary rules defined in order to partially check if the supplied information complies with its standard or recommended practice. The checking of whether the supplier is using competent agents to perform certain activities could be one such rule.

7 Model Validation

Our conceptual model is based on a systematic analysis of the IEC 61508 standard and on the authors' experience. To validate the usefulness of the model, we participated in a number of safety-certification meetings for a safety monitoring system in the maritime industry. From the issues raised by the certification body during these meetings, we randomly selected 30 and analyzed whether the information captured in our model could have either prevented or helped to address the issues. These issues could be classified in seven categories as shown in Table 1. Categories in the first five rows could be addressed by information collected based on the conceptual model. These categories represent 56% of the issues (17/30). The last two categories correspond to completeness issues and argumentation flaws and are not directly addressed by our model.

8 Related Work

Systematic development of safety cases (and more generally, dependability cases) is an important topic which is gaining increased attention in the dependability literature [3, 13, 21]. Kelly [14] provides an interesting and widely-accepted view on what a safety case is made up of. It divides a safety case into three parts: the safety requirements (or objectives), the

Table 1: Model Validation Findings

Type of Issue	Count
Missing traceability links	2
Missing requirement type e.g. performance, availability	2
Missing mode of operation for requirement	3
Unaddressed certifier expectations (e.g. use of particular notation or technique)	3
Unclear delineation of system blocks and interfaces	7
Unstated requirements, procedures , assumptions	6
Argumentation problems (redundancy, ambiguity, and reasoning issues)	7

argumentation, and the supporting evidence. The safety requirements are developed through various kinds of safety analyses (e.g., Fault Tree Analysis, and Failure Modes and Effects Analysis) and have been addressed extensively [9]. Building the argumentation in a safety case has been the focus of a lot of research in the past 15 years, e.g. [3, 7, 11, 14, 15, 18], with the Goal Structuring Notation (GSN) [14] as the basis for most of the work. However, there has been little research on providing a detailed characterization of the evidence underlying a safety case. What we presented in this paper is aimed to fill this gap for the software aspects of safety-critical systems.

The need for more effective collection and linking of safety evidence information has been noted before. In particular, Lewis [17] mentions the existence of a web of safety-relevant information covering not only the relationships between hazards and safety requirements but also between the operator of the system and operating procedures, the system functions and hardware elements, the system components and software tests, and so on. The conceptual model we developed in this paper provides a precise characterization of this web of information based on the IEC 61508 standard.

The sheer size and complexity of the documents comprising a safety case has been a constant challenge for safety assessors. The authors in [6, 17] propose the notion of electronic safety case, so that assessors can dynamically query a safety case for the information they need, instead of having to go through hundreds of pages of physical documents to find this information. As we discussed in Section 6.3, our conceptual model, when specialized to a particular context yields an information model for such electronic safety cases.

The authors in [5, 19] provide partial conceptualizations of IEC 61508, but they adopt a process-oriented view of the standard and thus focus on the processes involved when using the standard and assessing safety. The conceptual model we developed in this paper takes a much more holistic approach and captures all the key information concepts necessary to show compliance to the standard.

9 Conclusions

In this paper, we developed an extensible conceptual model, based on the IEC 61508 standard, to characterize the chain of safety evidence that underlies safety arguments about software.

We showed through some examples how our conceptual model can be specialized according to the needs of a particular context and described some important ways in which our model can facilitate software certification. An analysis of a random sample of issues raised in certification meetings showed that a majority of them would have been prevented or addressed by information collected according to our model. Applications of our model include: the precise specification of safety-relevant information requirements for system suppliers; defining a data model for developing a certification repository; the implementation of automatic, safety-relevant constraint verification (e.g., compliance with standard, recommended practice); and the automated generation of certification reports on demand. A detailed investigation of these activities and the development of appropriate tools to support them form facets of our future work.

Bibliography

[1] *OMG Object Constraint Language*.

[2] *Ships and marine technology – computer applications – general principles for the development and use of programmable electronic systems in marine applications (ISO 17894)*. International Organization for Standardization, 2005.

[3] P. BISHOP AND R. BLOOMFIELD, *A methodology for safety case development*, in Safety-Critical Systems Symposium, Springer, 1998.

[4] L. BRIAND, T. COQ, S. NEJATI, R. K. PANESAR-WALAWEGE, AND M. SABETZADEH, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, Tech. Report ModelMe! Technical Report 1, Simula Research Laboratory, `http://modelme.simula.no/`, 2009.

[5] P. CHUNG, L. CHEUNG, AND C. MACHIN, *Compliance flow - managing the compliance of dynamic and complex processes*, Knowledge-Based Systems, 21 (2008), pp. 332–354.

[6] T. COCKRAM AND B. LOCKWOOD, *Electronic safety cases: Challenges and opportunities*, in Current Issues in Safety-Critical Systems, in proceedings of Safety Critical Systems Symposium, Springer, 2003.

[7] G. DESPOTOU, D. KOLOVOS, R. PAIGE, AND T. KELLY, *Defining a framework for the development and management of dependability cases*, in Proceedings of 26th International System Safety Conference, System Safety Society, 2008.

[8] *Rules for classification of ships*. http://www.dnv.com/industry/maritime/rulesregulations /dnvrules/rulesclassships/, 2009.

[9] C. A. ERICSON, *Hazard Analysis Techniques for System Safety*, John Wiley & Sons, 2005.

[10] J. HOLT AND S. PERRY, *SysML for Systems Engineering*, Institute of Engineering and Technology, 2008.

[11] M. HUHN AND A. A. ZECHNER, *Analysing dependability case arguments using quality models*, in Proceedings of the 28th International Conference on Computer Safety, Reliability, and Security, SAFECOMP '09, Springer-Verlag, 2009, pp. 118–131.

[12] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[13] D. JACKSON, M. THOMAS, AND L. MILLETT, *Software for Dependable Systems: Sufficient Evidence?*, The National Academies Press, 2007.

[14] T. P. KELLY, *Arguing Safety – A Systematic Approach to Managing Safety Cases*, PhD thesis, University of York, 1998.

[15] T. P. KELLY AND J. MCDERMID, *Safety case patterns – reusing successful arguments*, in Proceedings of IEE Colloquium on Understanding Patterns and Their Application to System Engineering, 1998.

[16] T. P. KELLY AND R. A. WEAVER, *The goal structuring notation – a safety argument notation*, in Proceedings of the Dependable Systems and Networks - Workshop on Assurance Cases, 2004.

[17] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[18] J. A. MCDERMID, *Support for safety cases and safety arguments using sam*, Reliability Engineering and System Safety, 43 (1994), pp. 111 – 127.

[19] Y. PAPADOPOULOS AND J. MCDERMID, *A harmonised model for safety assessment and certification of safety-critical systems in the transportation industries*, Requirements Engineering, 3 (1998), pp. 143–149.

[20] D. STEINBERG, F. BUDINSKY, M. PATERNOSTRO, AND E. MERKS, *EMF: Eclipse Modeling Framework, 2/E*, Addison-Wesley Professional, 2009.

[21] S. P. WILSON, T. P. KELLY, AND J. A. MCDERMID, *Safety case development: Current practice, future prospects*, in Safety and Reliability of Software Based Systems - 12th Annual CSR Workshop, Springer-Verlag, 1997.

Paper II:
Using UML Profiles for Sector-Specific Tailoring of Safety Evidence Information

Using UML Profiles for Sector-Specific Tailoring of Safety Evidence Information

Rajwinder Kaur Panesar-Walawege[1,2], **Mehrdad Sabetzadeh**[1], **Lionel Briand**[1,2]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

Abstract:

Safety-critical systems are often subject to certification as a way to ensure that the safety risks associated with their use are sufficiently mitigated. A key requirement of certification is the provision of evidence that a system complies with the applicable standards. The way this is typically organized is to have a generic standard that sets forth the general evidence requirements across different industry sectors, and then to have a derived standard that specializes the generic standard according to the needs of a specific industry sector. To demonstrate standards compliance, one therefore needs to precisely specify how the evidence requirements of a sector-specific standard map onto those of the generic parent standard. Unfortunately, little research has been done to date on capturing the relationship between generic and sector-specific standards and a large fraction of the issues arising during certification can be traced to poorly-stated or implicit relationships between a generic standard and its sector-specific interpretation. In this paper, we propose an approach based on UML profiles to systematically capture how the evidence requirements of a generic standard are specialized in a particular domain. To demonstrate our approach, we apply it for tailoring IEC61508 – one of the most established standards for functional safety – to the Petroleum industry.

1 Introduction

Safety-critical systems are typically subject to safety certification, whose aim is to ensure that the safety risks associated with the use of such systems are sufficiently mitigated and that the systems are deemed safe by a certification body. A key requirement in safety certification is the provision of evidence that a system complies with one or more applicable safety standards. A common practice in defining standards for certification is to have a generic standard and then derive from it sector-specific standards for every industry sector that the generic standard applies to. The idea behind such a tiered approach is to unify the commonalities across different sectors into the generic standard, and then *specialize* the generic standard according to contextual needs. The generic standard is sometimes referred to as a *metastandard* [19].

A notable example in safety certification is the specialization of IEC61508 [7] – a generic standard that deals with the functional safety of electrical / electronic / programmable electronic safety-critical systems. In the process industry, this standard is adapted as IEC61511 [6], in railways as EN 50129 [5], in the petroleum industry as OLF070 [18], and in the automotive industry as the forthcoming ISO 26262 [8].

For specialization to be effective, it is important to be able to precisely specify how the evidence requirements stated in a generic standard map onto those stated in a derived standard. Unfortunately, there has been little work to date on systematizing the specification of the relationship between generic and sector-specific standards. This has led to a number of problems. In particular, Feldt et. al. [3] cite the lack of agreed-upon relationships between generic and derived standards as one of the main reasons behind certification delays, caused by ambiguities in the relationships and the need for subjective interpretations by the certification body and system supplier. Furthermore, Nordland [9] notes the lack of a well-formulated process for showing that a derived standard is consistent with a generic standard. This too is directly attributable to the lack of precise and explicitly-defined relationships between the standards.

In this paper, we propose a novel approach based on UML profiles [11] to capture the relationship between the evidence requirements of a generic standard and those of a sector-specific derivation. Briefly, our approach works by (1) building conceptual models for the evidence requirements of both the generic and sector-specific standards, (2) turning the conceptual model of the generic standard into a profile, and (3) using the profile for stereotyping the elements in the conceptual model of the sector-specific standard. Our approach offers two main advantages: First, it provides a systematic and explicit way to keep track of the relationships between a generic and a derived standard in terms of their evidence requirements. And second, it enables the definition of consistency constraints to ensure that evidence requirements are being specialized properly in the derived standard.

While the overall ideas behind our approach are general, we ground our discussions on a particular safety standard, IEC61508, and a particular derivation, OLF070 (used in the petroleum industry). On the one hand, this addresses a specific observed need in safety certification of maritime and energy systems; and on the other hand, it provides us with a concrete context for describing the different steps of our approach and how these steps fit together. The conceptual model characterizing the IEC61508 evidence requirements has been described in our earlier work [16]. The one for OLF070 has been developed as part this current work. Excerpts from both conceptual models will be used for exemplification

throughout the paper.

The remainder of this paper is structured as follows: In Section 2, we review background information for the paper. In Section 3, we describe our UML profile for IEC61508 and in Section 4 we discuss how the profile can be used for specialization of safety evidence. Section 5 compares our work with related work. Section 6 concludes the paper with a summary and suggestions for future work.

2 Background

In this section, we provide a brief introduction to safety certification (based on IEC61508), how safety evidence requirements can be structured through conceptual modeling, and UML profiles.

2.1 IEC61508-Based Certification

Safety-critical systems in many domains, e.g., the avionics, railways, and maritime and energy, are subject to certification. One of the most prominent standards used for this purpose is IEC61508. The standard sets forth the requirements for the development of electrical, electronic or programmable electronic systems containing safety critical components. This standard is concerned with a particular aspect of overall system safety, called functional safety, aimed at ensuring that a system or piece of equipment functions correctly in response to its inputs [7]. The standard defines requirements for hardware development, software development, and the development process that needs to be followed. The standard applies to systems with different required safety margins. This is encoded in the standard in the form of Safety Integrity Levels (SILs). The levels range from SIL 1 to SIL 4 and indicate the level of risk reduction measures that need to be in place based on the failure rate of the implementation and the acceptability of the risks involved. A number of sector-specific standards specialize IEC61508. These include IEC61511 in the process industry [6], EN 50129 [5] for railways, OLF070 [18] for the petroleum industry, and the upcoming ISO26262 [8] for the automotive industry.

2.2 Conceptual Modeling Of Compliance Evidence Information

In general, standards, irrespective of the domains they are targeted at, tend to be expressed as textual requirements. Since the requirements are expressed in natural language, they are subject to interpretation by the users of the standards. To make the interpretation explicit and develop a common understanding, we develop a conceptual model that formalizes the evidence requirements of a given standard. Such a model can be conveniently expressed in the UML class diagram notation [11].

For illustration, we show in Fig. 1 a small fragment of the conceptual model that we have built in our previous work on IEC61508 [16]. Concepts are represented as classes and concept attributes – as class attributes. Relationships are represented by associations. Generalization associations are used to derive more specific concepts from abstract ones. When an attribute assumes a value from a predefined set of possible values, we use enumerations. Finally, we use the package notation to make groupings of concepts and thus better manage the complexity.

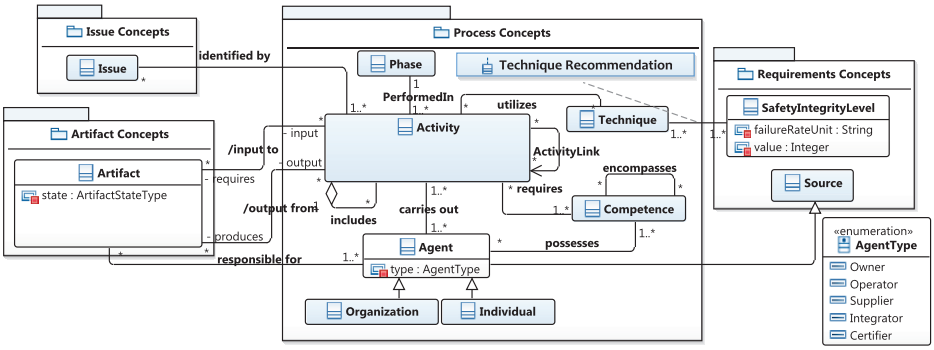


Figure 1: IEC61508 Process Concepts and Their Links

The diagram in Fig. 1 presents the concepts for describing the development process, packaged as `Process Concepts`, and how these relate to concepts in the `Issue Concepts`, `Artifact Concepts` and `Requirements Concepts` packages. From these other packages, we show only the concepts that related to those in `Process Concepts`. The central concept in the diagram of Fig. 1 is the notion of `Activity`, representing a unit of behavior with specific input and output. An activity can be further decomposed into sub-activities. A (life-cycle) phase is made up of a set of activities that are carried out during the lifetime of a system. Each activity utilizes certain techniques to arrive at its desired output, given its input. The selection of techniques is related to the safety integrity level that needs to be achieved. For example, if the activity in question concerns software verification, constructing formal proofs of correctness is usually unnecessary for low integrity levels, whereas, formal proofs are highly recommended for the highest integrity level. Each activity requires certain kind of competence by the agents performing it. The agent itself can be either an individual person or an organization. In either case, the agent is identified by the type of role it plays, for example the agent may be the supplier of a system or the operator. Agents can be made responsible for certain development artifacts. Further detail about the other packages shown can be found in [16].

2.3 UML Profiles

UML profiles [11] aim at providing a lightweight solution for tailoring the UML metamodel for a specific domain. The same mechanisms used by UML profiles for tailoring the UML metamodel can also be effectively used for tailoring standards compliance evidence according to domain-specific needs.

Briefly, UML profiles enable the expression of new terminology, notation and constraints by the introduction of context-specific stereotypes, attributes and constraints. Stereotypes are a means of extending a base metaclass. We extend the `Class`, `Property` and `Association` metaclasses, creating stereotypes for the concepts, their attributes and their relationships respectively. Moreover, constraints can be defined in a profile by using the Object Constraint Language (OCL) [1] to ensure that certain semantics are maintained in the new models to which the profile is applied. By using profiles the new models that employ the profile are still

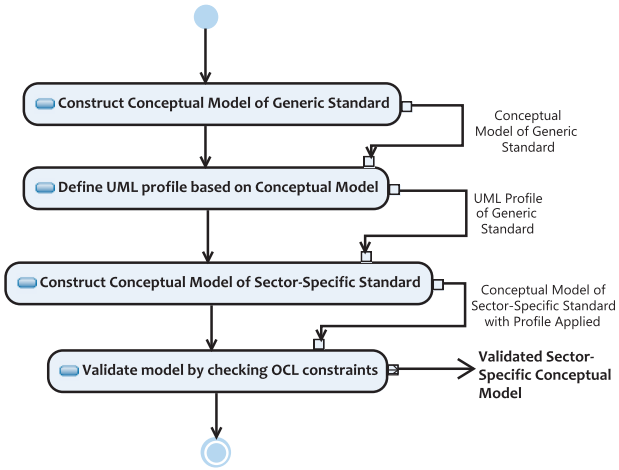


Figure 2: The Methodology for Specialization of a Generic Standard.

consistent with the UML metamodel.

As we describe in the subsequent sections, we use this mechanism to create a profile of the IEC61508 conceptual model (Section 3) and then use it to specialize the IEC61508 standard for the petroleum industry (Section 4).

3 UML Profile of the IEC61508 Standard

Our approach for specializing a generic standard is through the use of a UML profile. In Fig. 2, we show the methodology we propose for this purpose. The methodology consists of four main steps: (1) creating a conceptual model of the generic standard, we do this using a UML class diagram; (2) creating a UML profile based on the generic conceptual model; (3) creating a conceptual model of the sector-specific standard and applying the stereotypes from the UML profile of the generic standard; and (4) validating the OCL constraints of the profile over the sector-specific conceptual model to ensure that it is consistent with the generic standard. We apply this methodology for specializing the generic IEC61508 standard to the OLF070 standard for the petroleum industry.
Using profiles for specialization offers the following key advantages:

- We can incorporate the specific terminology used by a generic standard and still allow the use of context-specific terminology. For example, in IEC61508, we have the general concept of `ProgrammableElectronicSystem` (`PES`). OLF070 instead refers to very specific types of `PES`s in the petroleum industry, e.g., Fire and Gas system (F&G), Process Shut-Down system (PSD), Emergency Shut-Down system (ESD). These sector-specific concepts can all be stereotyped as `ProgrammableElectronicSystem` to capture the correspondence. It is of course possible to directly extend the conceptual model of a generic standard for a specific domain by adding new elements to it. However, this makes it hard to keep track of which concepts are from the generic standard and

which are from the sector-specific one. When a profile is used, all the stereotypes are known to be from the generic standard, hence a clear distinction is made between the terminologies.

- Stereotypes establish an explicit and rigorous mapping between the generic and sector-specific standards. This mapping can be used to ensure that, for a specific project, all the necessary evidence for demonstrating compliance has been collected. Further, the existence of such an explicit mapping makes it possible to define pairwise consistency rules between the generic and derived standards (using UML's rich constraint language, OCL), and to provide guidance to the users about how to resolve any inconsistencies detected.

As shown in Fig. 2, the basis of our profile of IEC61508 is the conceptual model of the IEC61508 standard. The process of creating a conceptual model of the evidence requirements of a given standard involves a careful analysis of the text of the standard. It requires skills in modelling, systems development and knowledge of the process of certification beyond merely reading the standard. To some extent, this can be viewed as a process of qualitative data analysis, where the data is the text of the standard and it is being analysed to identify from it, all the salient concepts and their relationships. This retrieved information from the text is used create a common understanding of the standard and as a means of explicitly showing the relationships that exist between the salient concepts.

We exemplify the process of creating the conceptual model of IEC61508 by showing an excerpt of the standard, and the concepts and relationships that have been gleaned from the excerpt. Fig. 3 shows a section of the IEC61508 standard that is dedicated to requirements applicable to the software of a safety-related system. In Fig. 3, we can see the salient concepts and relationships identified in the text - these have been highlighted by enclosing the relevant text in a box and numbering the identified section. Box 1 shows that the concepts `Phase` and `Activity` are of importance during the software development lifecycle (in Fig. 3 we have used the concept names shown earlier in Fig. 1). Box 2 identifies some key relationships between phases and activities. An activity is performed during a phase and has specified inputs and outputs. Box 3 indicates that a generic life cycle is prescribed by the standard while not precluding deviations in terms of phases and activities. Box 4 presents the concepts: technique, safety integrity level and techniques recommendation - indicating that activities utilize certain techniques based on the safety integrity level. The same concepts and relationships may be found in several places in the standard. Once the text has been marked up in this manner, a glossary is created to ensure that consistent terms are used to refer to the same concepts and relationships. A part of this glossary, describing the most important concepts is shown in Table 1. The conceptual model is created from this set of concepts and their relationship and serves as the metamodel of the profile.

Fig. 4 shows a bird-eye's view of the different packages that make up the metamodel for our IEC61508 UML profile. The packages contain abstractions for modelling of the main concepts of IEC61508. We briefly explain each package. For more details, see [16]. The `System Concepts` package describes the breakdown of the system at a high level; the `Hazard Concepts` package contains the abstraction for describing the hazards and risks for the system; the `Requirements Concepts` package for the different types of requirements, including safety requirements; the `Process Concepts` package for describing the development process

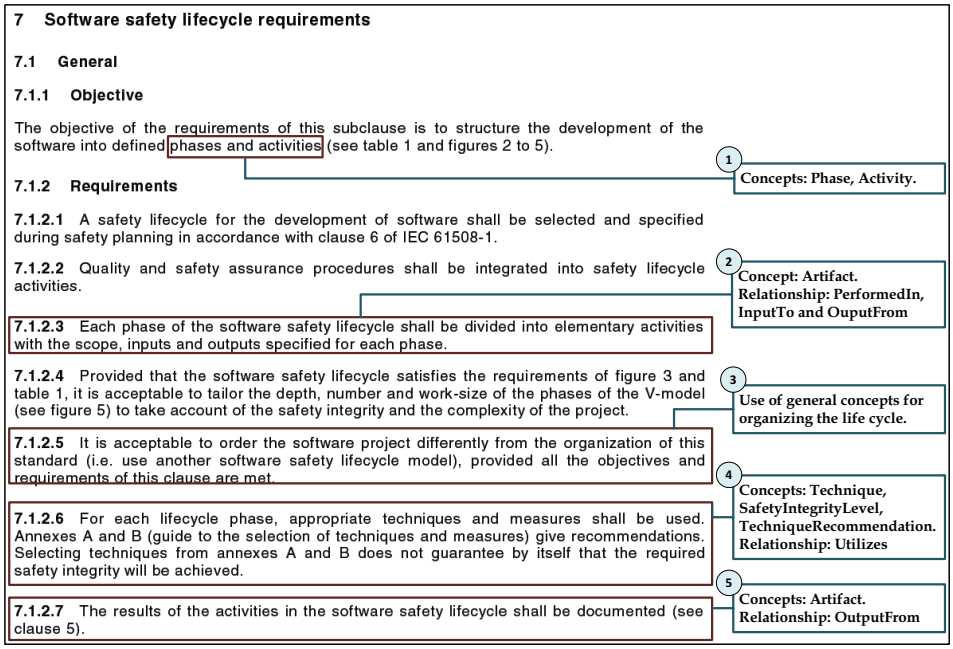


Figure 3: An Excerpt of IEC61508 showing the textual source of some of the Process elements

(details given in Section 2.2); the `Artifact Concepts` package for describing the different types of artifacts created as supporting evidence; the `Guidance` package for describing the other standards and recommended practices that will be used to develop the system, the `Issue Concepts` package for describing the defects or enhancements that may have given rise to changes; the `Configuration Management Concepts` package for describing the unique versions for all the components that make up the system, the `Justification Concepts` package to capture the assumptions and rationale behind the various decisions that are made during development; and the `Domain-Specific Concepts` package for capturing the enumerations for concept attributes in other packages (e.g., requirement type, system operating mode). The elements of the conceptual model are mapped almost directly into the profile. The concepts become stereotypes that extend the metaclass `Class`, the relationships become stereotypes that extend the metaclass `Association` and the attributes of these two extend the metaclass `Property`.

Table 1: Description of Main Concepts from the IEC61508 Metamodel

Stereotype	Description
`Activity`	A unit of behaviour in a process.
`Agent`	A person or organization that has the capability and responsibility for carrying out an activity.
`Artifact`	One of the many kinds of tangible by-products produced during the development of a system.
`Assumption`	A premise that is not under the control of the system of interest, and is accepted as true without a thorough examination. Assumptions can, among other things, be related to the environment of the system, the users, and external regulations.
Continued on next page ...	

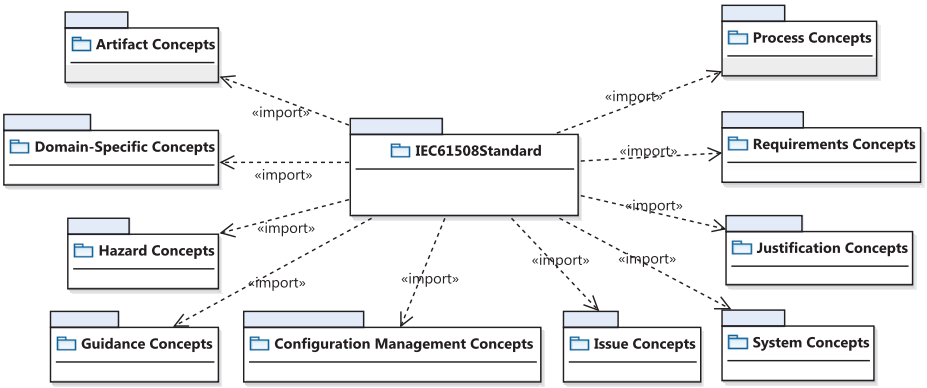


Figure 4: Packages of the IEC61508 Metamodel

Continued from previous page ...	
Stereotype	**Description**
`Block`	Entity of hardware or software, or both, capable of accomplishing a specified purpose.
`Change`	A modification made to the PES, Block or Artifact.
`Competence`	The ability to perform a specific task, action or function successfully.
`ControlledItem`	A PES, Block or Artifact for which meaningful increments of change are documented and recorded.
`Defect`	An error, failure, or fault in a system that produces an incorrect or unexpected result, or causes it to behave in unintended ways.
`Description`	A planned or actual function, design, performance or activity (e.g., function description).
`DesignatedState`	The state of the EUC related to safety, the EUC is either in a safe state or an unsafe state.
`Diagram`	Specification of a function by means of a diagram (symbols and lines).
`Enhancement`	Provision of improved, advanced, or sophisticated features.
`Error`	Discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition.
`Event`	A single occurence in a series of occurences that cause a hazard to occur.
`Failure`	Termination of the ability of a functional unit to perform a required function.
`Fault`	Abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function.
`GeneralStandard`	A standard that provides generic recommendations on a specific subject to a number of related domains.
`HardwareBlock`	Any entity of hardware – this may be mechanical, electrical or electronic that is used in the composition of the system.
`HazardousElement`	The basic hazardous resource creating the impetus for the hazard, such as a hazardous energy source such as explosives being used in the system.
`Hazard`	Any real or potential condition that can cause injury, illness, or death to personnel damage to or loss of a system, equipment or property or damage to the environment.
`Individual`	Refers to a person.
`InitiatingMechanism`	The trigger or initiator event(s) causing the hazard to occur. The IM causes actualization or transformation of the hazard from a dormant state to an active mishap state.
`Instruction`	Specifies in detail the instructions as to when and how to perform certain jobs (for example operator instruction).
`Interface`	An abstraction that a block provides of itself to the outside. This separates the methods of external communication from internal operation.
`Issue`	A unit of work to accomplish an improvement in a system.
`List`	Information in a list form (e.g., code list, signal list).
`Log`	Information on events in a chronological log form.
`Mistake`	Human action or inaction that can produce an unintended result.
`NonProgrammable-HardwareBlock`	Electro-mechanical devices (electrical) solid-state non-programmable electronic devices (electronic).
Continued on next page ...	

Continued from previous page ...	
Stereotype	**Description**
OperatingMode	The different modes that a system can be operating in, e.g. normal, maintenance, test, emergency.
Organization	A social arrangement which pursues collective goals, which controls its own performance, and which has a boundary separating it from its environment.
Phase	A set of activities with determined inputs and output that are carried out at a specific time during the life of a system.
Plan	Explanation of when, how and by whom specific activities shall be performed (e.g., maintenance plan).
Programmable-ElectronicSystem	System for control, protection or monitoring based on one or more programmable electronic devices, including all elements of the system such as power supplies, sensors and other input devices, data highways and other communication paths, and actuators and other output devices.
Programmable-HardwareBlock	Any physical entity based on computer technology which may be comprised of hardware, software, and of input and/or output units.
Rationale	The fundamental reason or reasons serving to account for something.
RecommendedPractice	Sound practices and guidance for the achievement of a particular objective.
Report	The results of activities such as investigations, assessments, tests etc. (e.g., test report).
Request	A description of requested actions that have to be approved and further specified (e.g., maintenance request).
Requirement	A necessary attribute in a system; a statement that identifies a capability, characteristic, or quality factor of a system in order for it to have value and utility to a user.
ResidualRisk	Risk remaining after protective measures have been taken.
Risk	Combination of the probability of occurrence of harm and the severity of that harm.
SafeState	The state of the EUC when safety is achieved.
SafetyIntegrity-Level	The probability of a safety-related system satisfactorily performing the required safety functions under all the stated conditions within a stated period of time.
SafetyRequirement	A prescriptive statement that ensures that the system carries out its functions in an acceptably safe manner.
SectorSpecific-Standard	A standard that provides recommendations for a specific industrial sector (e.g., the energy sector).
SoftwareBlock	Any entity of software that may be used for controlling the system – this may be embedded or application software or even different levels of software such as module, component, subsystem, system.
SoftwareLevel	The different levels into which a software system can be decomposed, e.g. System, subsystem, component and module.
Source	An abstract concept that can represent a person, organization or standard that can be a source of requirements to a system.
Specification	Description of a required function, performance or activity (e.g., requirements specification).
Standard	An established norm or requirement, typically provided as a formal document that establishes uniform engineering or technical criteria, methods, processes and practices.
Technique-Recommendation	A particular technique recommended based on the safety integrity level of the requirements that have been allocated to the block in question.
Technique	A procedure used to accomplish a specific activity or task.
UnsafeState	The state of the EUC when safety is compromised.
UserRole	An aspect of the interaction between a PES and the human elements.

Our IEC61508 profile consists of:

- 57 stereotypes that extend the metaclass `Class`, used to characterize the evidence elements
- 53 stereotypes that extend the metaclass `Association`, used to characterize the traceability links amongst the various evidence elements.
- 6 stereotypes extend the metaclass `Property`, used on the role names of the corresponding associations.

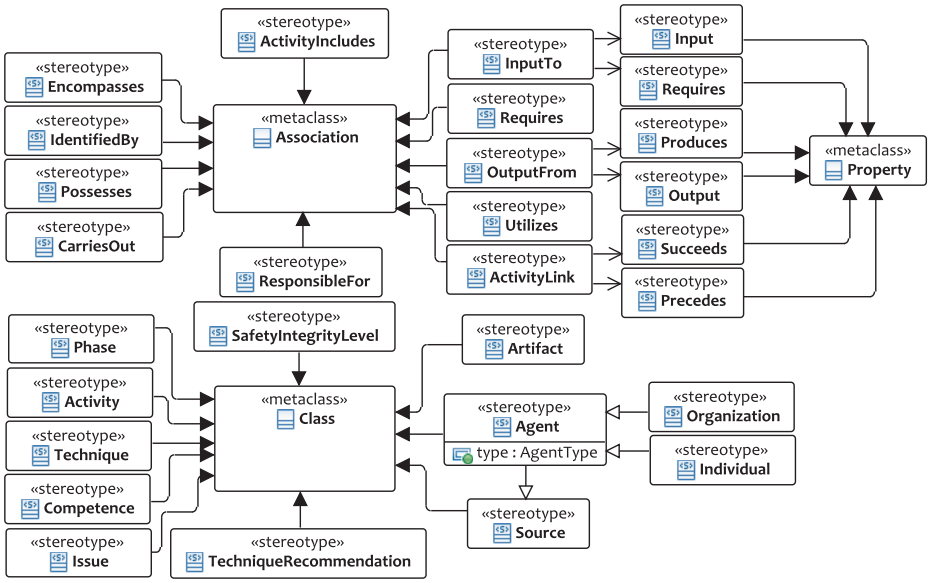


Figure 5: IEC61508 Profile Fragment for the System Development Process

Besides these stereotypes, stereotypes extending the `Class` and `Association` metaclasses have OCL constraints to ensure they are used consistently. We will discuss these constraints and provide examples later in this section.

Since the profile is quite large and cannot be fully explained in this paper, as an example, in Fig. 5, we show the stereotypes created to manage the development process. These are the stereotypes derived from the partial conceptual model shown in Fig. 1. The IEC61508 standard does not mandate a specific development life-cycle such as the waterfall or iterative lifecycle; it does however state that a number of specific activities should be carried out. We have the stereotype `Activity` to model this. An `Activity` can itself include other sub activities and this is modelled by the association stereotype `ActivityIncludes`. Certain activities may precede or succeed others and this is modelled via the association stereotype `ActivityLink` along with its properties `Precedes` and `Succeeds`.

In safety-critical systems, it is very important to ensure that all work is carried out by personnel with the required knowledge and skills. IEC61508 mandates that this information be part of the compliance evidence. Hence, for each activity, we model both the required competence and that of the agent performing the activity via the stereotypes `Agent` and `Competence` along with `CarriesOut`, `Requires` and `Possesses`. An activity may have certain artifacts that are needed in order to carry it out and it will produce certain artifacts upon its completion. These concepts are modelled using the stereotypes `Artifact`, `InputTo`, `OutputFrom`, `Requires`, `Produces`, `Input` and `Output`. Finally, each activity will use certain techniques to create its output. These techniques are chosen based on the level of safety required and hence we have the stereotypes `Technique` and `TechniqueRecommendation`.

As stated earlier, there are OCL constraints for the class and association stereotypes. These

Table 2: OCL Constraints on Stereotypes

Stereotype	Constraint
CarriesOut	self.base_Association.memberEnd->select(p:Property | not (p.class.getAppliedStereotype('IEC61508Profile::Activity').oclIsUndefined()))->size()=1 and self.base_Association.memberEnd->select(p:Property | not (p.class.getAppliedStereotype('IEC61508Profile::Agent').oclIsUndefined()))->size()=1
OutputFrom	self.base_Association.memberEnd->select(p:Property| not (p.class.getAppliedStereotype('IEC61508Profile::Activity').oclIsUndefined()))->size()=1 and self.base_Association.memberEnd->select(p:Property| not (p.class.getAppliedStereotype('IEC61508Profile::Artifact').oclIsUndefined()))->size()=1
Activity	1: self.base_Class.ownedAttribute->collect(c:Property | c.association)->select(a:Association | not a.getAppliedStereotype('IEC61508Profile::OutputFrom').oclIsUndefined())->size()>0 2: self.base_Class.ownedAttribute->collect(c:Property|c.association)->select(a:Association | not a.getAppliedStereotype('IEC61508Profile::CarriesOut').oclIsUndefined())->size()>0

constraint enforce the structural consistency of the evidence information in the sector-specific derivations. Specifically, for any association stereotyped with X, we must check that the endpoints of the association are stereotyped correctly according to the endpoints of X in the profile metamodel. For example, consider the `CarriesOut` stereotype. We need a constraint to ensure that any association with this stereotype connects two elements stereotyped `Agent` and `Activity`, respectively. This constraint is shown in Table 2. A similar constraint is shown for `OutputFrom`, to ensure that any association having this stereotype has endpoints that are stereotyped `Artifact` and `Activity`.

For stereotypes extending the `Class` metaclass, we need to verify that any stereotyped element respects the multiplicity constraints of the profile metamodel. We show an example in Table 2: we have constraints to ensure that an element with the `Activity` stereotype is linked to at least one element with the `Artifact` stereotype and at least one element with the `Agent` stereotype.

The profile only needs to be created once per standard, and then can be reused for specializing the generic standard to any number of domains. Once the profile is created, the stereotypes of the profile are applied to the conceptual model of the domain-specific standard, also expressed as a UML class diagram. For the derived standard there are three things to bear in mind to ensure its consistency with the generic standard: (1) which concepts will be used directly from the generic standard (possibly with different terminology), (2) which concepts are specific to the domain and thus new, and (3) which concepts, from the generic standard, have been deliberately left out as they may not be applicable to the domain, in which case this omission is clearly noted and explained. The conceptual model of the derived standard is created in a manner similar to the generic standard, except that, the profile stereotypes are applied and the OCL constraints are checked to enforce the semantics of the specialization and guide the user in creating a structurally sound information model for a derived standard.

4 Specializing IEC61508 for the Petroleum Industry

OLF070 is a derivation of IEC61508, elaborating the safety concerns that are specific to control systems in the petroleum industry. We discuss at a high level how OLF070 refines IEC61508. Recall the packages shown in Fig. 4: the `Artifact Concepts`, the `Configuration Management Concepts`, the `Issue Concepts`, the `Guidance Concepts`, and the `Justification Concepts` are the same in OLF070 as in IEC61508. The `Hazard Concepts` are the same, apart from the fact that in OLF070, the most common hazards have been defined in the standard already. The change in the `System Concepts` is that in addition to specifying the breakdown of the system, a particular component can be specified as either being part of a local safety function (e.g., process shutdown) or a global safety function (e.g., emergency shutdown). The `Requirements Concepts` specify that the SIL level of most common components can be obtained from a table provided in the standard unless there is a deviation in the component from what is described in the standard, in which case the SIL level is calculated using the procedures specified by IEC61508. The `Process Concepts` and the `Domain-Specific Concepts` are different in that there are specific processes and specific terminology used in the petroleum industry for developing the systems. In this section, we illustrate the specialization process by showing how the profile described in the previous section can be used for tailoring the evidence required by the OLF070 standard [18].

To preserve the continuity of our examples from the previous section, we focus on the development process aspects of OLF070, and more precisely on one of the phases envisaged in the standard, called the Pre-Execution Phase. This phase is concerned with developing a Plan for Development and Operation (PDO) of an oilfield. The PDO contains the details of all the systems that need to be created to make the oilfield functional. The phase ends with the creation of the PDO document that is then sent to the authorities to get permission for the project and used to select the main engineering contractor. In this phase, a number of activities are carried out: (1) all the equipment to be installed at the field and all the safety instruments systems (SIS) are defined; (2) hazards are identified; (3) a risk analysis is performed to gauge the extent of the risks that need to be mitigated; (4) safety functions (such as fire detection, gas detection, process shut-down) and the safety integrity levels are specified based on the results of the risk analysis.

In Fig. 6, we present a small excerpt of the OLF070 conceptual model and show the concepts we have just described as the different activities that take place during the Pre-Execution Phase. The stereotypes from our IEC61508 profile have already been applied. The phase is documented in the artifact called `PlanForDevelopmentAndOperation`. This is in compliance with IEC61508, whereby each phase should have a plan documenting it. For some of the activities, we show the relevant inputs and outputs and the agents that need to perform them. We use the stereotypes from our IEC61508 profile to show how this OLF070 model excerpt relates to IEC61508. Some of the stereotype we have already explained in Section 3. The four new ones here are `DocumentedIn` for the result of a phase, `BasedOn` to show whether an artifact is based on a standard, `Standard` to indicate a type of material used to create an artifact and `PerformedIn` for indicating which phase an activity is performed in. Note that stereotypes can have attributes, e.g the attribute `type` for the stereotype `Agent`, shown in Fig. 6, has the value `Owner` to indicate that the Safety Engineer is employed or commissioned by the owner of the system to be developed. For linking to artifacts and facilitating navigation to

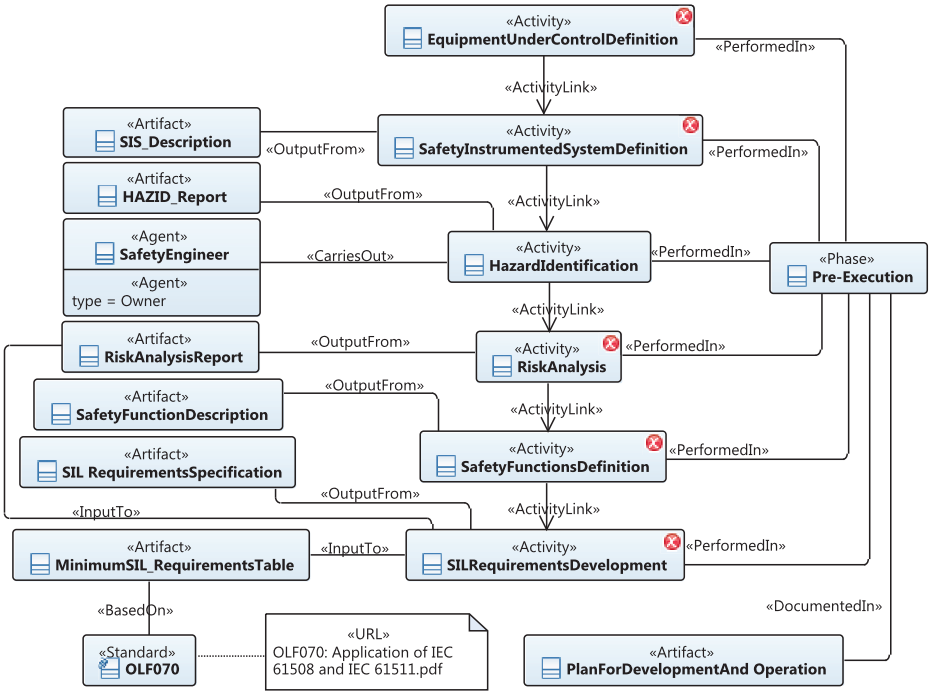


Figure 6: An example phase from OLF070

them, we can include URLs and file references in the conceptual model. An example is shown in the figure, where we link the OLF070 element to the actual document for the standard.

As discussed in Section 3, we use OCL constraints for enforcing consistent use of the profile. Once the stereotypes have been applied to the modelled elements, we can validate the model using an OCL checker, e.g. the Rational Software Architect OCL tool [4] that we use here. In Fig. 6, we can see that five of the elements have a red cross in their upper right-hand corner. These element have failed the OCL validation. The errors generated are shown in Fig. 7. The first five errors concern the constraint that an activity should have an agent performing it. The model elements `EquipmentUnderControlDefinition`, `SafetyInstrumentedSystemDefinition`, `RiskAnalysis`, `SafetyFunctionsDefinition`, and `SILRequirementsDevelopment` do not have a corresponding agent element. For `EquipmentUnderControlDefinition`, a further constraint has been violated: there is no output specified from that activity, indicated by the last error in the snapshot of Fig. 7. Thus, in addition to providing a means to explicitly show the relationships between the generic and sector-specific standard, the profile enables users to check whether the requirements of the generic standard are maintained in the sector-specific one.

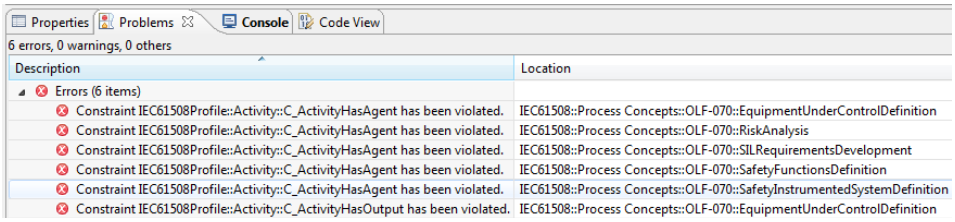


Figure 7: Error Report showing violated OCL Constraints

5 Related Work

Using UML profiles to adapt UML to a specific context is very common. The Object Management Group have so far standardized three profiles: the UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE) [13], the UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms (QFTP) [12], and the UML Profile for Schedulability, Performance and Time (SPT) [15]. All three include safety-relevant concepts. However, in contrast to our work, none of these were designed for characterizing the evidence required for compliance to safety standards.

Zoughbi et. al. [20] propose a UML profile for the RTCA DO-178B standard[17] used in commercial and military aerospace software. This profile enables software engineers to directly add certification information to software models. The concepts modeled are targeted at addressing a major requirement of RTCA DO-178B having to do with traceability between requirements and design and eventually code. This information together with evidence of other quality assurance activities would form the basis of full compliance to the standard. The approach we propose in this paper differs from [20] in the following ways: Firstly, we focus on a different and broader standard; secondly, our profile includes a wide range of concepts related to the management of the development process in safety-critical systems, whereas [20] deals primarily with requirements and design; and thirdly and most importantly, we use profiles as a basis for sector-specific specialization – specialization is not tackled in [20].

The Software Assurance Evidence Metamodel (SAEM) [14] is a proposal from the OMG, concerned with managing assurance evidence information. A main distinction between our work and SAEM is that we aim at characterizing the evidence that needs to be collected for certification based on a standard. Instead, SAEM is standard-independent and mainly directed towards linking the evidence to claims and the evaluation of the claims in light of the evidence. An abstract specification of evidence such as the one given by SAEM will therefore need to be complemented with an evidence conceptual model for a specific standard, e.g., our IEC61508 conceptual model. Indeed, just as we use profiles for specializing IEC61508 for a specific sector, one can use profiles to incorporate SAEM into the conceptual model of a given standard and create a metamodel that captures both the evidence requirements for compliance, and also the evaluation of whether the evidence is sufficient to substantiate the claims.

Chung et. al. [2] study the problem of compliance of a user-defined workflow with the activities envisaged in IEC61508. Their approach is to check (process) compliance by comparing user-defined activities in an organization against models of the activities in the standard. Our work is close to [2] in its goal to model compliance information; however, we

go beyond the process aspects of IEC61508 and provide an evidence information model for the entire IEC61508, which can in turn be specialized to sector-specific needs through the use of profiles.

6 Conclusion and Future Work

In this paper we presented a methodology for ensuring that a generic standard can be specialized in a systematic manner for a particular domain. We do this by capturing the generic standard as a conceptual model using a UML class diagram and use this as a basis for creating a UML profile. The profile is then applied to the conceptual model of a sector-specific standard and used as an explicit means of keeping track of the relationships between the two. We exemplify our methodology by showing excerpts of the IEC61508 conceptual model that we have created, the UML profile based on this model and how we apply this profile to a conceptual model of the OLF070 standard which is a sector-specific derivation of IEC61508 for the petroleum industry.

Our approach offers two key benefits: (1) It incorporates the specific concepts used by a generic standard into the sector-specific standard whilst making a clear distinction between the two; and (2) It explicitly captures the mapping between two standards and defines consistency rules between them, which can be automatically verified and used for providing guidance to the users about how to resolve any inconsistencies.

Having established a means to capture the evidence required for a specific standard, we are now working on a means to create instantiations of these conceptual models such that we can create repositories of evidence for safety certification. Subsequently, we plan to carry out case studies to assess the cost-effectiveness of our methodology in the context of certification. Another prime concern is the ability to certify a system to multiple and often overlapping standards. For example, in the petroleum industry, it is quite common to certify a system to both OLF070 and to one of the NORSOK standards such as the NORSOK I-002 for Safety Automation Systems [10]. In future work, we plan to extend our methodology so that we can express how a repository of evidence information addresses each standard in a collection of inter-related standards. Finally, to aid the certification process from the perspective of a certification body, we would like to extend our work to the evaluation of evidence as proposed by the SAEM. This would lay the groundwork for a complete certification infrastructure based on industry standards.

Bibliography

[1] *OMG Object Constraint Language*.

[2] P. CHUNG, L. CHEUNG, AND C. MACHIN, *Compliance flow - managing the compliance of dynamic and complex processes*, Knowledge-Based Systems, 21 (2008), pp. 332–354.

[3] R. FELDT, R. TORKAR, E. AHMAD, AND B. RAZA, *Challenges with software verification and validation activities in the space industry*, in ICST'10, 2010, pp. 225–234.

[4] *IBM Rational Software Architect*. `http://www.ibm.com/developerworks/rational/products/rsa/`.

[5] *Railway Applications âĂŞ Safety-related electronic railway control and protection systems.*, 1999.

[6] *Functional safety - safety instrumented systems for the process industry sector (IEC 61511).*, 2003.

[7] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[8] *Road vehicles – functional safety*, 2009. ISO draft standard.

[9] O. NORDLAND, *A critical look at the cenelec railway application standards*. `http://home.c2i.net/odd_nordland/~SINTEF/tekster/A_critical_look_at_rail_standards.htm`, 2003.

[10] *Safety and automation system (SAS)*, 2001.

[11] *UML 2.0 Superstructure Specification*, August 2005.

[12] *UML profile for modeling quality of service and fault tolerance characteristics and mechanisms specification*. http://www.omg.org/spec/QFTP/1.1/, 2008.

[13] *UML profile for modeling and analysis of real-time and embedded systems (marte)*. http://www.omg.org/spec/MARTE/1.0/, 2009.

[14] *Software Assurance Evidence Metamodel (SAEM)*. http://www.omg.org/spec/SAEM/, 2010.

[15] O. M. G. (OMG), *UML profile for schedulability, performance and time*. http://www.omg.org/spec/SPTP/, 2006.

[16] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[17] *DO-178B: Software considerations in airborne systems and equipment certification*, 1982.

[18] *Application of IEC61508 and IEC61511 in the Norwegian Petroleum Industry*, 2004.

[19] M. UZUMERI, *Iso 9000 and other metastandards: Principles for management practice?*, Academy of Management Executive, 11 (1997).

[20] G. ZOUGHBI, L. BRIAND, AND Y. LABICHE, *Modeling safety and airworthiness (RTCA DO-178B) information: conceptual model and uml profile*, Software and Systems Modeling, (2010), pp. 1–31.

Paper III:
A Model-Driven Engineering Approach to Support the Verification of Compliance to Safety Standards

A Model-Driven Engineering Approach to Support the Verification of Compliance to Safety Standards

Rajwinder Kaur Panesar-Walawege[1,2], **Mehrdad Sabetzadeh**[1], **Lionel Briand**[1,2]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

Abstract:

Certification of safety-critical systems according to well-recognised standards is the norm in many industries where the failure of such systems can harm people or the environment. Certification bodies examine such systems, based on evidence that the system suppliers provide, to ensure that the relevant safety risks have been sufficiently mitigated. The evidence is aimed at satisfying the requirements of the standards used for certification, and naturally a key prerequisite for effective collection of evidence, is that the supplier be aware of these requirements and the evidence they require. This often proves to be a very challenging task because of the sheer size of the standards and the fact that the textual standards are amenable to subjective interpretation. In this paper, we propose an approach based on UML profiles and model-driven engineering. It addresses not only the above challenge but also enables the automated verification of compliance to standards based on evidence. Specifically, a profile is created, based on a conceptual model of a given standard, which provides a succinct and explicit interpretation of the underlying standard. The profile is augmented with constraints that help system suppliers with establishing a relationship between the concepts in the safety standard of interest and the concepts in the application domain. This in turn enables suppliers to demonstrate how their system development artifacts achieve compliance to the standard. We illustrate our approach by showing how the concepts in the domain of sub-sea control systems can be aligned with the evidence requirements in the IEC61508 standard, which is one of the most commonly used certification standard for control systems.

1 Introduction

Safety-critical systems are often subject to a stringent safety certification process, aimed at providing assurance that a system is deemed safe by a certification body. Increasingly, system suppliers are asked by such bodies to provide their justification for the safe operation of a system in the form of a *safety case*, which provides well-reasoned arguments based on the collected safety evidence that the overall safety objectives of a system are being met [14]. One approach to certification is to demonstrate, based on evidence, compliance to a safety standard, such as IEC61508 for Programmable Electronic Systems (PES) [11].

Verifying compliance to safety standards proves to be a very challenging task because of their sheer size and the fact that they are mostly textual and subject to subjective interpretation. On the supplier side, they run the risk of missing critical details that need to be recorded *during* system development. This means that they will have to reconstruct the missing evidence after the fact. Doing so is often very expensive, and the outcomes might be far from satisfactory. On the certifier side, poorly structured and incomplete evidence often leads to significant delays and loss of productivity, and further may not allow the certifier to develop enough trust in the system that needs to be certified. It is therefore very important to devise a systematic approach, which is amenable to effective automated support, to specify, manage, and analyze the safety evidence used to demonstrate compliance to standards.

Motivated by the above challenges, we have studied in our previous work different facets of the problem of safety evidence specification and management. Specifically, we proposed an approach to specify safety evidence using conceptual modeling [24] and a technique for tailoring generic evidence requirements according to sector-specific needs (e.g., in the railways, avionics, and maritime and energy sectors) [23]. A recurring theme in this earlier work is the use of standard Model-Driven Engineering (MDE) technologies, such as UML [19] and OCL [1] for specification, storage, and analysis of safety evidence information.

In this paper, drawing on the same MDE principles underlying our previous work, we develop a novel approach for assisting system designers in relating the concepts of their application domain to the evidence requirements of the standards that apply to the domain. The research is motivated by a natural and real need that we have observed in software safety certification. Specifically, the majority of the evidence artifacts that the suppliers record are based on the supplier's concepts for the application domain, as opposed to the concepts of the certification standards. The absence of an explicit and precise link between the two conceptual frameworks can pose two main challenges. First, the certifier may not be able to comprehend the evidence, and second, it becomes very difficult to verify whether the evidence collected using the domain concepts is covering all the evidence aspects mandated by the standard.

To give a concrete example, in the IEC61508 standard, a Programmable Electronic System (PES) is the system for controlling or monitoring one or more programmable electronic devices, including all elements of the system such as sensors, communication paths, and actuators. It has software that is used to send commands for controlling the various different types of equipment. A sub-sea control system on the other hand, is made up of a Sub-sea Control Module (SCM) that incorporates a Sub-sea Electronics Module(SEM). The SCM executes the commands for opening or closing valves that control the oil well. These commands are sent from the Sub-sea Control Unit (SCU) which is software that run on the oil rig in what is called the Topside Processing Unit(TPU).[2, 12]. In this scenario, the certifier

needs to know which is the PES, and which is the software system. The PES in this case is the SCM and the software controlling and monitoring it, is the SCU. The correlation of these simple pieces of information provides clarification to the certifier who needs to understand the system being certified.

To address the above problem, we propose a novel technique that guides system designers in establishing a sound relationship between the domain model for a safety-critical application and the evidence model for a certification standard. Our approach makes use of UML profiles. This enables us to build upon mature MDE technologies and tailor them for our specific needs, particularly for specifying and automatically checking the constraints that must hold for compliance with safety standards.

More precisely, we begin with developing a profile based on the conceptual model of a given standard. The profile is then augmented with verifiable constraints that help system suppliers to systematically relate the concepts in the standard to the concepts in the application domain. The resulting relationship provides a clear route for the supplier to demonstrate how their development artifacts can be used for showing compliance to the standard. We illustrate our approach by showing how the concepts in the domain of sub-sea control systems can be related to the evidence requirements in the IEC61508 standard, which is one of the most commonly used certification standards for control systems.

The remainder of this paper is structured as follows: In Section 2, we review background information for the paper and in Section 3 we outline our overall approach for creating certification evidence for compliance. In Section 4, we present our UML profile for IEC61508 and in Section 5 we discuss how the profile can be used for the creation of certification evidence. Section 6 compares our work with related work. Section 7 concludes the paper with a summary and suggestions for future work.

2 Background

In this section, we briefly introduce safety certification and how the evidence for standards compliance can be structured through conceptual modeling, and UML profiles.

2.1 Safety Certification

Safety-critical systems are typically subject to a safety certification process. The aim of certification is to provide assurance that the system has been deemed safe for use in a specific environment. This is usually carried out by a third-party certification body. The certification is usually based on a specific standard applicable to the domain in which the system is operated, e.g., there is the general standard IEC61508 for the certification of electrical, electronic or programmable electronic systems that are used in safety-critical environments, the IEC61511 standard for the process industry [10], EN50129 [9] for railways, and NORSOK I-002 [18] for safety automation systems in the petroleum industry. All these standards present requirements for how the system should be created to ensure the quality of the end product and more specifically to ensure that the system is safe for operation. The justification for safe operation of a system is usually presented as a safety case [14].

A safety case is made up of three principal parts [14]: safety objectives, arguments, and

evidence. Demonstrating the satisfaction of the objectives involves gathering evidence during the lifecycle of a system and constructing well-reasoned arguments that relate the evidence to the objectives. With the growing use and complexity of software in safety-critical systems, licensing and safety regulatory bodies increasingly require system suppliers to provide an explicit software safety case. A software safety case is a part of an overall system safety case, which provides assurance that the software elements of a system satisfy the safety aspects stated in the technical and software requirements specification of the system [17]. While the argumentation aspects of software safety cases have been studied before [14], little has been done to precisely specify the evidence that underlies software safety arguments [13]. As a result, suppliers of safety-critical software have been left without proper guidance on what evidence to collect during development. This has led to the suppliers having to recover the relevant evidence after the fact, which can be extremely costly or even infeasible. In addition, the quality of the overall safety case is bound by the quality of the weakest link. Hence, current practices for managing software safety evidence can severely limit the effectiveness of safety cases in general. In this paper, we provide a flexible approach for systematically specifying safety evidence and establishing a precise link between the evidence and the requirements of relevant standards. This will in turn help with the construction of more definitive software safety cases.

2.2 Conceptual Models

In general, standards, irrespective of the domains they are targeted at, tend to be expressed as textual requirements. Since the requirements are expressed in natural language, they are subject to interpretation by the users of the standards. To make the interpretation explicit and develop a common understanding, we propose the development of a conceptual model that formalizes the evidence requirements of a given standard. Lewis [16] expresses the need for presenting a safety case as an information model. He highlights the need for creating a formal structure for a safety case and the need to present the relationships that exist between atomic items of information resulting in a web of information that supports the safety case argument. In order to represent these relationships as required by a particular standard, we create a conceptual model that allows us to represent the main factors that need to be considered for certification and the relationships amongst them. The fundamental elements that we need to represent are 1) concepts, 2) attributes, 3) inter-concept relationships and 4) constraints. Additionally, as standards can be quite large, it is useful to have a means to divide the concepts into useful groupings. The UML [19] class diagram notation can be used to conveniently express the conceptual model. Concepts are represented as classes and concept attributes – as class attributes. Relationships are represented by associations. Generalization associations are used to derive more specific concepts from abstract ones. When an attribute assumes a value from a predefined set of possible values, we use enumerations. Finally, we use the package notation to make groupings of concepts and thus better manage the complexity [24].

2.3 UML Profiles

UML profiles [19] are a lightweight solution for extending the UML metamodel for a specific domain. They enable the expression of new concepts, notation and constraints by the intro-

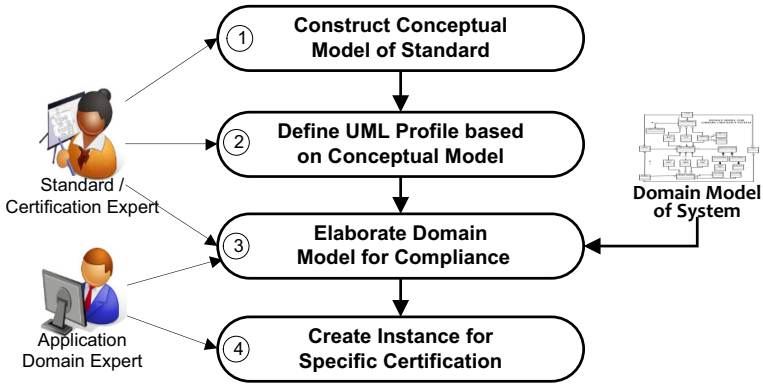


Figure 1: Methodology for the Creation of Evidence of a Safety Standard.

duction of context-specific stereotypes, attributes and constraints. Stereotypes are a means of extending a base metaclass of the UML metamodel. We extend different metaclasses to create stereotypes for the concepts, their attributes and their relationships respectively. Moreover, constraints can be defined in a profile by using the Object Constraint Language (OCL) [1] to ensure that certain semantics are maintained in the models to which the profile is applied. By using profiles the models that employ the profile are still consistent with the UML metamodel. The profile stereotypes along with constraints written in OCL provide us with the mechanism to guide the creation of the evidence requirements for a specific standard.

As we describe in the subsequent sections, we use this mechanism to create a profile of the IEC61508 conceptual model (Section 4) and then use it to create the evidence required for the certification of a sub-sea control system in the petroleum industry.

3 Approach Overview

We propose a approach for assisting system suppliers in preparing for certification of their systems according to industry-relevant standards. Our approach guides system developers in establishing a relationship between a domain model of a safety-critical application and the evidence model of a certification standard. We make use of UML profiles which allows us to build upon mature MDE technologies for specifying and automatically checking the constraints that must hold for compliance with safety standards. The approach consists of four main steps as shown in Figure 1, which will be illustrated by our case study.

The first step is the creation of a conceptual model of the standard according to which a system needs to be certified. This process involves interpreting the text of the standard and picking the main concepts presented in it, any attributes the concepts may have, and any inter-concept relationships. Note that making these concepts and their relationships explicit is an important aspect of compliance [5, 13, 16].

The second step is the creation of a UML profile based on the conceptual model. The profile is in turn used for stereotyping the elements of a domain model of the system to be certified. Broadly, a domain model is a representation of the core concepts in an area of

interest. In this paper, we use the term domain model to refer to concepts that represent the physical and abstract components of a *family* (class) of systems in a particular application area (e.g., sub-sea control systems), the environment in which this family of systems function, and the key artifacts built throughout development. An example of product family [25] is a Fire and Gas Protection system that will consist of sensors being used to detect fire or combustible gas, a controller that does processing based upon the input from the sensors and then deploys certain actuators such as sprinklers or dampers. This is a generic description of a class of systems – each variant of the system will have very specific types of sensors and actuators with specific actions that should take place upon the detection of fire or gas. Following the norm in MDE, we assume domain models are represented as UML class diagrams[15]. Using a profile makes it possible to establish a concrete link between the evidence requirements of a given standard and a domain model. In this paper, we do not concern ourselves with the construction of domain models. Good references and guidelines already exist [15].

When a stereotype from the profile of a given standard is applied to a domain model element, it shows how that element fulfills the requirements from the standard. The profile is created by mapping the concepts in the conceptual model as extensions of the metaclass 'Class' in the UML metamodel, the attributes of the concepts are made into attributes of the class to which the stereotypes are applied to, the relationships between the concepts are mapped as extensions of the metaclass 'Association'. Enumerations are used for describing either standard-specific or user-specific data types. OCL constraints are added to the stereotypes to ensure certain properties of the stereotypes as well as to guide system developers in elaborating the domain model for the system being developed.

The third step requires the elaboration of the domain model. Specifically, elaboration means the application of the profile stereotypes to the appropriate domain model elements, and refining the domain model so that it satisfies the OCL constraints of the stereotypes. These refinements could include the addition of new domain model elements or making changes to the existing ones (e.g., adding new attributes, revising multiplicities).

The process starts by applying a stereotype to the domain model itself, stating which standard the domain model needs to comply with. If the standard of interest is IEC61508, the stereotype could be `IEC61508Model`. The OCL constraints associated with this stereotype will start the guidance process for augmenting the domain model with other stereotypes. This in turn may require the domain model to be updated so that the stereotype constraints are satisfied. Each new stereotype applied will have further constraints that will need to be satisfied for the model to be valid. This chain of constraints will guide the elaboration of the domain model, so that it will cover all aspects that are necessary for certification. Ultimately, that elaborated domain model will represent a precise specification for the safety evidence and explicit links to the standard's requirements.

Finally, for certifying a specific system (variant) from a product family, step four in Figure 1 is performed. This step creates an instantiation of the UML class diagram representing the elaborated domain model. In other words, an object diagram of the domain model is built to represent the specific properties of a system variant.

Steps one, two and three need input from an expert who understands the certification process and the standard to which the system will be certified; whereas, in steps three and four the knowledge of an application domain expert is required. Finally, we note that steps one and two are carried out once per standard; step three and the creation of the domain model is

done once per product family; and step four is performed once for each variant that is subject to certification.

We exemplify this whole process by creating a conceptual model and then profile of the IEC61508 standard. We present the profile in Section 4 and show how the domain model of a sub-sea control system is elaborated with the application of stereotypes from the IEC61508 profile in Section 5.

4 The IEC61508 Profile

4.1 IEC61508 Standard

The IEC61508 standard presents requirements to facilitate the development of safety-related electrical/electronic/programmable electronic systems (E/E/PES). The goal of the standard is to ensure the functional safety of safety-related E/E/PES systems. Functional safety is a component of overall safety, for example, the activation of an alarm in response to a fire detection by a control system is a functional safety measure, whereas the use of fire resistant walls to control the spread of fire is not, whilst it is still a part of overall safety measures. A function that a control system performs to ensure that the system remains in a safe state is referred to as a safety function. Each safety function specifies which safety objective is to be achieved (safety function requirement) and the level of integrity with which the safety function is implemented (safety integrity level).

To achieve the required level of safety, the standard recommends the use of a safety lifecycle. The lifecycle should contain certain activities such as a hazard analysis and risk assessment to determine the hazards that can occur and the risks that they pose. Together, these activities determine what has to be done to avoid hazardous situations (derivation of safety requirements) and the level to which safety has to be provided (derivation of safety integrity levels). The derived safety requirement are allocated to either certain functions of a designated E/E/PE safety-related system, other technology safety-related systems, or to external risk reduction facilities. The IEC61508 standard is only concerned with the allocations made to the E/E/PE system. Once the requirements have been allocated, the realization of the system begins for both the hardware and software aspects of the E/E/PE system. The activities concerned with the installation and commissioning, operation and maintenance, and the final overall safety validation of the system begin alongside the realization of the system.

4.2 IEC61508 Conceptual Model

The conceptual model for the IEC61508 standard was built in our previous work on IEC61508 [24]. As mentioned in Section 2, we use UML class diagrams to create the conceptual model where concepts are represented as classes and concept attributes as class attributes; relationships are represented by associations. When an attribute assumes a value from a predefined set of possible values, we use enumerations. Finally, we use the package notation to make groupings of concepts and thus better manage the complexity.

The conceptual model has a total of 10 packages, containing abstractions for modelling the main concepts of IEC61508. We briefly explain each package. For more details, see [24]. The `System Concepts` package describes the breakdown of the system and reflects both hardware

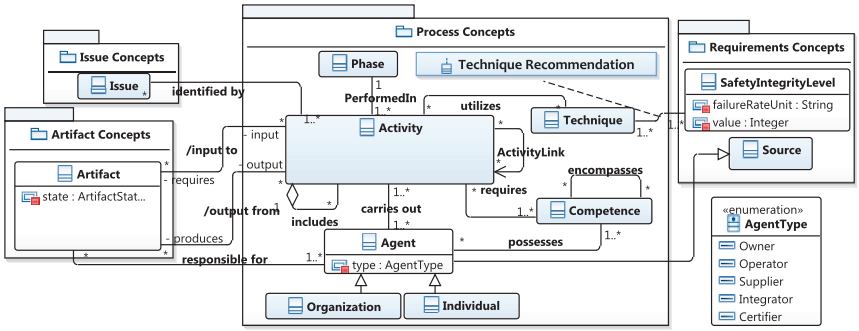


Figure 2: Process Concepts Package of the IEC61508 Conceptual Model

and software concepts; the `Hazard Concepts` package captures the abstraction for describing the hazards and risks for the system and leads to the specification of safety requirements; the `Requirements Concepts` package captures the requirements for creating, operating, maintaining and decommissioning control systems; the `Process Concepts` package is for describing the development process for creating the system; the `Artifact Concepts` package is for describing the different types of artifacts created as supporting evidence during the development of the system; the `Guidance` package is for describing the other standards and recommended practices that will be used to develop the system, the `Issue Concepts` package is for describing the defects or enhancements that may give rise to changes; the `Configuration Management Concepts` package is for describing the unique versions for all the components that make up the system, the `Justification Concepts` package to capture the assumptions and rationale behind the various decisions that are made during development; and the `Domain-Specific Concepts` package for capturing the enumerations for concept attributes in other packages (e.g., requirement type, artifact state).

As a small example, we show in Figure 2 part of the conceptual model for specifying the process of development. The main concept in this package is the concept of activity, this is a unit of work which has specific artifacts as defined inputs and outputs. An activity can be decomposed into further activities and is performed within a larger unit of work called a phase. A phase defines a means to manage a related set of activities together and a number of phases are used to manage the development of the entire system. An activity may have one of more agents that perform it and there are certain techniques that are utilized to carry it out. The techniques selected for an activity are based on the safety integrity level that needs to be achieved. Thus, module testing alone may be sufficient for a low level of safety integrity but at higher levels, testing along with formal proofs may be required. The agents that carry out the activity need to possess the competence that the activity requires and may be either individuals or organizations. The concepts related to the process are together used to show that competent agents have created the system in an organized manner. Details for the other packages can be found in [24].

4.3 IEC61508 Profile

The IEC61508 profile is a means of showing how a system fulfills the requirements of the IEC61508 standard. The profile is based on a conceptual model of the standard that was created in our earlier work [24]. From the conceptual model we capture all the concepts and their relationships in terms of stereotypes and use OCL constraints for capturing some of the more complex requirements of the standard. The stereotypes are used to highlight how a particular aspect of the system fulfils the IEC615508 standard.

The IEC61508 profile consists of the following:

- 1 stereotype that extends the metaclass `Model`, that characterizes the base domain model as being certified to the IEC61508 Standard..
- 4 stereotypes that extend the metaclass `Package`, that is used to organize the evidence at a high level.
- 54 stereotypes that extend the metaclass `Class`, that are used to characterize the evidence elements.
- 53 stereotypes that extend the metaclass `Association`, that are used to characterize the relationships between the evidence elements.
- 6 stereotypes extend the metaclass `Property`, that are used on role names of certain associations.

All stereotypes have documentation attached to them that explains what the stereotype is meant for and OCL constraints that perform two main functions: they ensure that the stereotypes are used in the way intended, and provide guidance to the user as to which stereotypes need to be used in the first place, e.g., the stereotype `Activity` has constraints that ensure that elements with the stereotype `Agent` also exist in order to show who is performing the activity. In this way we create the web of evidence information mentioned in Section 2. We use OCL constraints for a number of purposes:

(i) To ensure that mandatory aspects of the standard are accounted for.

(ii) To ensure the correct type of stereotypes at the two ends of associations.

(iii) To ensure that elements with certain stereotypes are connected to other specific elements.

(iv) To ensure that elements with certain stereotype have specific properties - this helps when creating instances of the model.

(v) To help with the creation of user-defined enumerations defined in the conceptual model.

As it would be difficult to show all stereotype of the profile within the size constraints of this paper, in Figure 3 we show a fragment of the profile corresponding to the process concepts package discussed earlier along with the stereotypes applied at the model and package level. We also use this fragment to show examples of the five types of constraints mentioned above.

As we stated earlier, the IEC61508 profile is meant to be applied to a domain model of the system to be certified. The first stereotype to be applied is at the model level: the

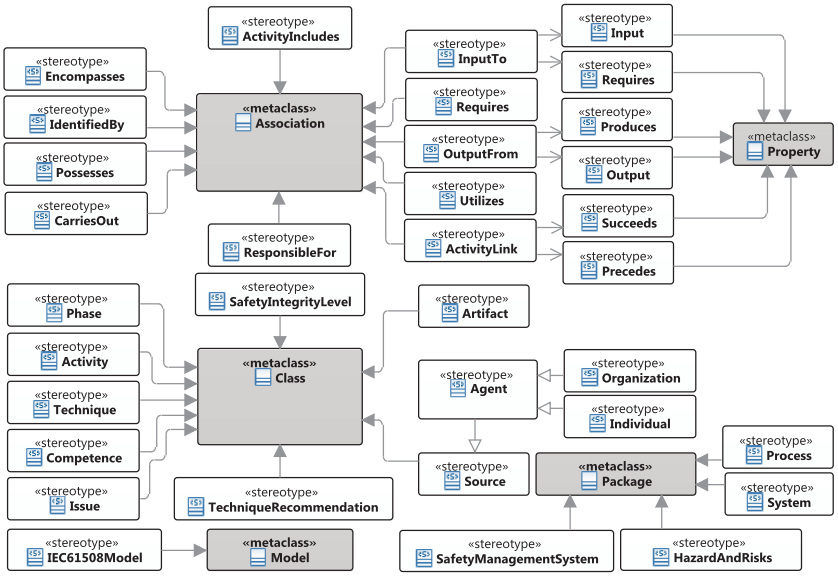


Figure 3: IEC61508 Profile Fragment for the System Development Process

`IEC61508Model` stereotype, which starts the incremental guidance process about which types of evidence to create. This stereotype has attached to it the OCL constraints that ensure that at a minimum four specific packages exist in the model with the stereotypes `Process`, `System`, `SafetyManagementSystem` and `HazardsAndRisks` (constraints of type 1). As an example, we show the constraint on this stereotype for ensuring that the `HazardsAndRisks` stereotype exists on a UML package in the model (for the sake of brevity we have omitted the name and context of the constraints shown):

```
self.base_Model.allOwnedElements()->
  exists(e | e.oclIsTypeOf(uml::Package) and
    not e.getAppliedStereotype(
      'IEC61508Profile::HazardsAndRisks').oclIsUndefined())
```

The keyword `self` refers to the element being constrained, in this case the `IEC61508Model` stereotype. Properties and attributes of an element are referenced using the dot notation. The `base_Model` reference is used to access the model to which the stereotype has been applied, in this case the domain model. The `allOwnedElements` is an operation that returns all the elements in the model. `Exists` is an OCL operation that will check that least one element in a collection of elements satisfies the given constraint. The constraint in the `exists` clause specifies that at least one element is of type `Package` using the operation `oclIsTypeOf` and that this element also has the stereotype `HazardsAndRisks` applied to it (using the operation `getAppliedStereotype`).

The rationale behind requiring these packages comes from the IEC61508 standard. The IEC61508 standard advocates a risk-based approach for determining the required level of

safety measures for safety-relevant systems. Hence the need for the `HazardsAndRisks` package. Risks can only be determined based upon the hazards that will exist when the system is used, thus it is important to have a breakdown of the system, bearing in mind both the hardware and software aspects of the system as well as the role of human users. This breakdown will be kept in the `System` package. The standard also put emphasis on having clearly specified technical and management activities and a clear identification of all responsible persons within the organization that perform these activities. The management information is kept in the `SafetyManagementSystem` package whereas the technical activities are specified within a safety life-cycle and kept in the `Process` package.

The standard does not require a specific kind of life-cycle but does state which activities should be carried out and which artifacts should be produced. Thus, we have the stereotype `Phase` to model the life-cycle and the stereotype `Activity` to model the activities. The `Process` package has a constraint (type 1) on it that specifies that it should contain in it elements with the stereotype of `Phase`:

```
self.base_Package.allOwnedElements()->
  exists(el:Element | el.oclIsKindOf(uml::Class) and
    not(el.getAppliedStereotype('IEC61508Profile::Phase').
      oclIsUndefined()))
```

The `Phase` stereotype has a constraint attached to it that states that every phase must have at least one `Activity` defined for it (a type 3 constraint). This means that there must be elements that have the stereotype `Activity` in the same package and be attached to the element with the stereotype `Phase`. This is done through two different constraints, one on the class stereotype `Phase` and the other on the association stereotype `PerformedIn` (see Figure 2). On the stereotype `Phase`, we have a constraint that states that there should be an association with a stereotype `PerformedIn` originating from the element that has this stereotype:

```
self.base_Class.ownedAttribute->
  collect(c:Property | c.association)->
    select(a:Association | not a.getAppliedStereotype(
      'IEC61508Profile::PerformedIn').
        oclIsUndefined())->size()>0
```

On the stereotype `PerformedIn`, there is the constraint that states that this stereotype can only be applied to an association that is between a pair of elements that have the stereotypes `Activity` and `Phase`, respectively (a type 2 constraint):

```
self.base_Association.memberEnd->
  select(p:Property not (p.class.getAppliedStereotype(
  'IEC61508Profile::Activity').oclIsUndefined()))->
    size()=1
and
self.base_Association.memberEnd->
  select(p:Property| not (p.class.getAppliedStereotype(
    'IEC61508Profile::Phase').oclIsUndefined()))->
    size()=1
```

An activity can include sub activities or it can be linked to another activity by either preceding or succeeding it, all these relationships are modelled by the stereotypes `ActivityIncludes` and `ActivityLink`, along with its properties `Precedes` and `Succeeds`. Activities are to be performed by competent agents using recommended techniques, that are based upon the safety integrity level allocated to a component. All these aspects are modelled using the stereotypes `Agent`, `Requires`, `Competence` along with `CarriesOut`, `Possesses`, `Technique` and `TechniqueRecommendation`. An activity may require certain artifacts as input and upon completion produce certain artifacts as outputs. The stereotypes `Artifact`, `InputTo`, `OutputFrom`, `Requires`, `Produces`, `Input` and `Output` are used to model these concepts. Constraints on the stereotype `Activity` ensure that for every activity, the agent that carried out the activity is defined as well as the output from the activity. Constraints are also used to create properties for the elements on which stereotypes have been applied or for creating user-defined types, e.g., An element with the `Artifact` stereotype applied to it should have a property called 'State' of type 'ArtifactStateType' which is a user defined enumeration (this constraint combines both type 4 and 5 constraints):

```
self.base_Class.ownedAttribute->
  one(p:Property | p.name='State' and
    p.type.name='ArtifactStateType' and
      p.type.oclIsTypeOf(uml::Enumeration))
```

These types of constraints allow the user to define domain-specific values for the enumeration, the profile only gives the name and type of the property. An advantage of using OCL constraints is that they can be automatically checked using any OCL validation engine, thus providing a means of efficiently checking large amounts of evidence in terms of completeness and consistency. The stereotypes together with the constraints defined on the stereotype guide the user in creating an information model of the evidence necessary for certification. This in turn will enable the systematic collection and automated analysis of evidence. In the next section, we show how the IEC61508 profile can be used to manage the certification evidence for a sub-sea control system in the petroleum industry.

5 Case Study: Application of the IEC61508 Profile to the Sub-sea Control Domain

To validate our approach, we have applied it for guiding the construction of a domain model for sub-sea control systems in compliance with IEC61508, and to partially create the evidence for a specific variant of the system. Our sub-sea domain concepts were defined in close consultation with experts in a large maritime and energy company and based on a reading of the relevant literature where the architecture and the components of sub-sea systems (including the control software operating on them) are described [2, 12, 18, 27]. Due to space constraints, we are unable to show the entire domain model or to go through all the guidance steps provided by the profile. Instead, we will focus in this section on a fragment of the domain model, shown in Figure 4, and illustrate how our proposed approach is applied in a concrete way over this fragment.

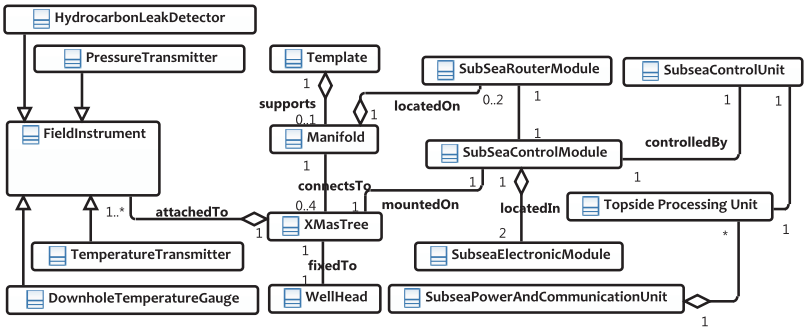


Figure 4: A Domain Model Fragment of a Sub-Sea Control System

In a sub-sea system, the wellhead attaches to the sub-sea oil or gas well and interfaces to the drilling and other production equipment housed in what is known as a Christmas Tree (CT). A CT in this context is an assembly of control valves, pressure gauges, and chokes put on the top of a well to control the flow of oil and gas once the well drilling operation has been completed. The CT controls the flow of oil or gas to the manifold which provides the connections to direct the oil or gas away from the production system. All this equipment is anchored to the seabed via a structural frame called the template. Mounted on the CT is the Sub-sea Control Module (SCM) that receives commands from the Sub-sea Control Unit (SCU) that is executing in the Topside Processing Unit (TPU) located in the Sub-sea Power and Communication Unit (SPCU). The SCM also sends signals from the sub-sea instruments to the SCU. The signals are sent from the SCM via the Sub-sea Router Module (SRM) to a router in the SPCU that passes the signal to the TPU. A more complete description of these components can be found in [2, 12].

Once the initial domain model has been created, the `IEC61508Model` stereotype is applied to the domain model and the OCL constraints of this stereotype are validated. Figure 5 shows the beginning of the guidance process for creating the evidence. The first thing required is the creation of four packages that have the stereotypes: `Process`, `System`, `SafetyManagementSystem` and `HazardsAndRisks`. The packages themselves can be named using the supplier's own terminology, but the specified stereotypes need to be applied. Once these stereotypes have been applied, the next set of (failed) constraints will provide guidance on what stereotypes to apply next.

Figure 6 shows that five constraints have failed once the package stereotypes have been applied. Hazards have not yet been identified in the `HazardsAndRisks` package; phases have not yet been identified in the `Process` package; agents and their competence have not been identified in the `SafetyManagementSystem` package, and blocks have not been identified in the `System` package. As an example, we will show the application of the stereotypes that identify the system components.

The SCU controls and monitors the sub-sea wells through the operator station, so the stereotype `SoftwareBlock` is applied to this element. The system being controlled is the SCM that contains the Sub-sea Electronic Module (SEM) that links to the different instruments. Thus,

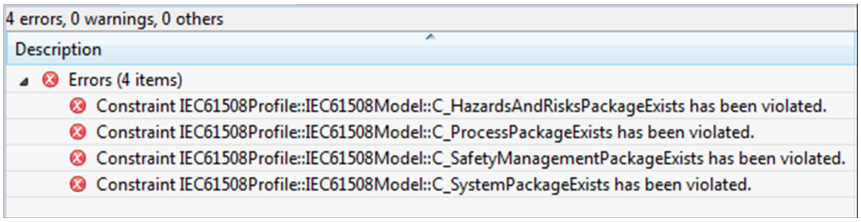


Figure 5: Error Report Showing the Violated OCL Constraints of the IEC61508Model Stereotype

the stereotype `ProgrammableElectronicSystem` is applied to the SCU and the stereotype `ProgrammableHardwareBlock` to the rest of the elements. If we now validate the model then, we get new constraints that are violated as the model is now missing further information.

In Figure 7, all the elements have a cross in their upper right-hand corner. These element have failed the OCL constraint validation. For brevity, we show in Figure 8, the errors generated for the SCU only; similar errors are also generated for the other elements.

All system blocks need to have requirements allocated to them. In this model, the allocated requirements have not been added to the model, leading to the violation of constraint regarding requirements. Blocks also need to have unique identifiers, which have not been yet added to the elements – in the petroleum industry, every components of a system has a unique identifier called a tag. The standard also recommends version control of the system components - a version attribute has not yet been added to the elements either, thus the violation of that constraint. If we add the elements to satisfy these constraint violations, then we get the model depicted in Figure 9. Two requirements at the system level have been added to the model - SReq1 and SReq2. They model two different kinds of requirements that are common on sub-sea control systems: the shutdown of parts of the system due to an emergency and the monitoring of the status of certain instruments. All requirements are kept in an artifact called the SystemSoftwareRequirements.

A user-defined enumerated type has also been added. A constraint on the `SoftwareBlock` stereotype requires the need to show the decomposition level of the software. The constraint specifies the name of the attribute ('Level') and the type ('Enumeration'). The actual literal values are set by the user as relevant to their industry. In this case the literals used are 'System', 'FunctionModule' , 'LogicModule' and 'Driver' - this is the breakdown that is most commonly used in the sub-sea industry and hence the user is able to use appropriate domain terminology. Attached to each stereotype is documentation that can help the user to understand how to use a particular stereotype, as shown in Figure 10. Thus all stereotypes are documented in this way to aid the user in elaborating the domain model with the necessary stereotypes. The documentation is useful is giving general guidance as the constraints alone would not be sufficient. The creation of the profile and the models and the validation of constraints can be performed using a UML modelling tool, e.g., Rational Software Architect [8]. We used this tool in our case study here.

As more stereotypes are added and the constraints are evaluated, the web of evidence is created for a particular product family. Once the elaboration of the domain model is complete with all the generic information, the instance for a particular system variant can be created. In Figure 11, we show a small instance model conforming to the domain model that we present in

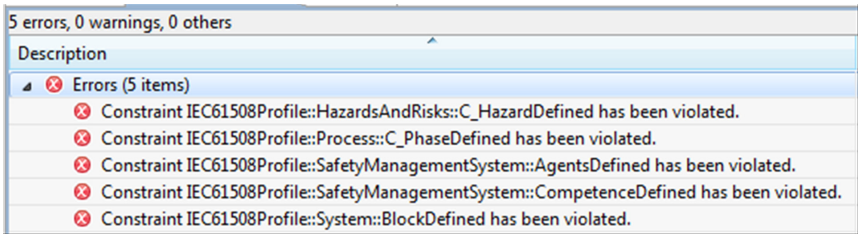


Figure 6: Error Report Showing the Violated OCL Constraints After Application of the Package Stereotypes

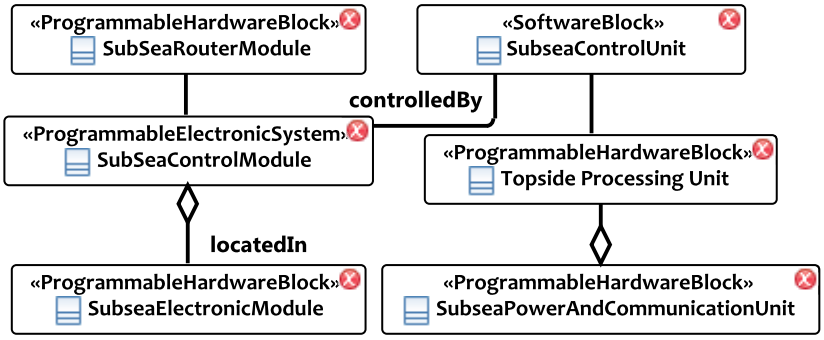


Figure 7: Fragment of the Domain Model After Application of the System Stereotypes

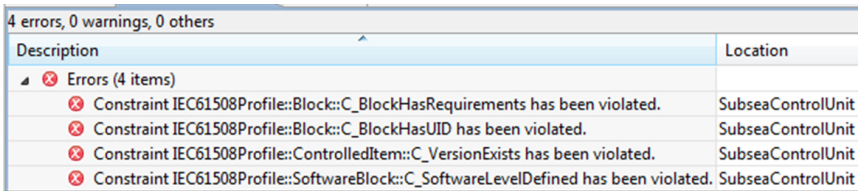


Figure 8: Error Report Showing the Violated OCL Constraints for the Subsea Control Unit

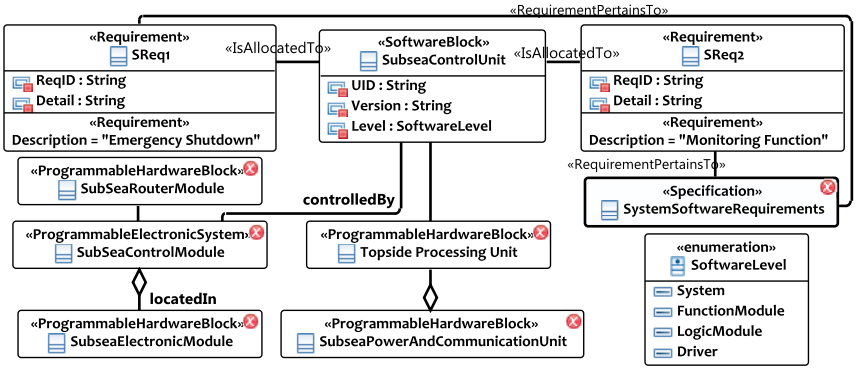


Figure 9: Fragment of the Domain Model After Application of the System Stereotypes

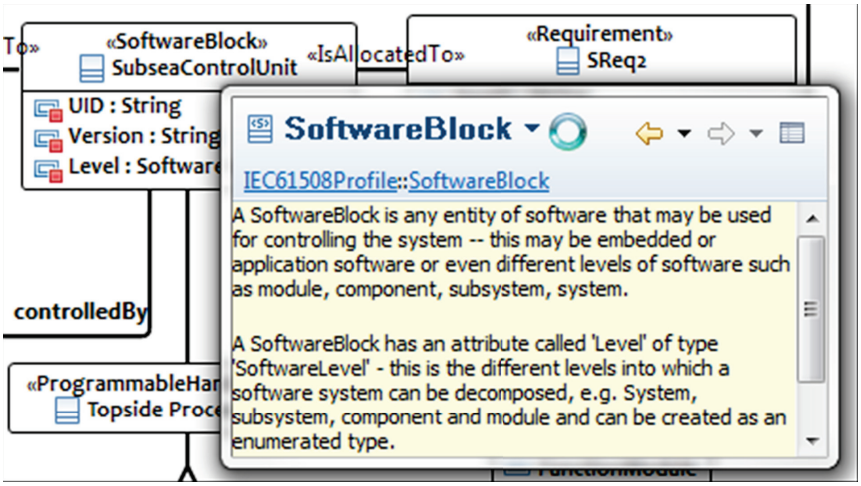


Figure 10: Documentation for a Stereotype

Figure 9. In this particular instance there is one template and one manifold. The manifold has two CT structures on it, each connected to a wellhead. There are two SCMs, both controlled by a single SCU. The unique Id given to the SCU is 'T1823a', the version of the software for the SCU is 1.2 and this is the system level software version. For hardware equipment, the version would store the model number or serial number of the piece of equipment. Three requirements are shown: RQ1.121 and RQ1.212 are instances of SReq1 which is the requirement in the domain model concerning emergency shutdown of the system. There are two instances in the actual system of this requirement to deal with the shutdown of the two wells independently. A further requirement, RQ1.1 is shown that would be for monitoring the status of some piece of equipment - there would be multiple instances of this requirement in the system to monitor the various instruments. All requirements are kept in the specific requirements artifact called theABC_OilField Requirements. The UID, Version, Level, ReqID and Detail attributes were added to the elements due to constraints on the stereotypes, this allows specific values to be set for these elements in the instance model. The stereotypes can have attributes as well, as in the case of 'Description' for the `Requirement` stereotype. The value for this stereotype is set in the domain model and does not change in the instance model. The use of an elaborated domain model for the certification evidence of a product family and the instance model for the evidence for a particular variant allows a high degree of reuse and helps to reduce the effort needed in creating the evidence for each particular variant.

6 Related Work

There are two areas of related work that are important to mention regarding our work: the creation of electronic safety cases and the use of UML for the development of safety-critical systems.

The need for constructing electronic safety cases has been identified by [6] and [16] in order

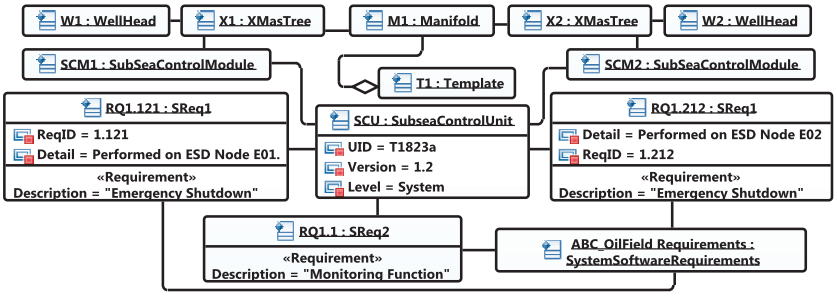


Figure 11: Instance Model Created from the Elaborated Domain Model

to manage the complexity and large amounts of information that needs to be kept for a safety case. Lewis [16] calls for an underlying information model to manage the complex links that exist between the various pieces of safety evidence. We propose an automated approach to do so, in the context of the IEC61508 standard. Our conceptual model provides the underlying information model for the standard and our profile provides a practical mechanism for using this information in order to create the relevant artifacts for a safety case. Cockram and Lockwood [6] present a hypertext linked approach to linking all the pieces of information for a safety case in the form of a proprietary tool. We on the other hand use model-driven engineering technologies, and in particular UML profiles, to aid with the creation of the safety case for a particular standard. The profile can be exported and used in any UML modelling tool. This allows one to keep a set of inter-related information items which can be viewed directly from the tool, or can be transformed into different views by the use of reporting tools.

The use of model-based technologies is gaining pace in industry. Especially, UML is increasingly used in the development of safety-critical software. The Object Management Group (OMG) have standardized the UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE) [21] and the UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms (QFTP) [20]. Both these profiles are used for modelling the real-time and performance properties of safety-critical systems. Similary, Berkenkotter [3] and Hannemann [4] have created a profile for the railway domain that aids the design and verification of interlocking functionality. However, neither of these are meant to characterise the evidence requirements of a standard to which safety-critical systems are certified.

A profile that deals with certain aspect of certification is proposed by Zoughbi et. al. [28]. Their profile enables the direct addition of certification information to software models, for compliance with the RTCA DO-178B standard [26] used in commercial and military aerospace software. However, this profile is targeted at maintaining traceability between requirements, design and code, which is a part of the requirements of the RTCA DO178B standard. The profile that we propose deals with a different standard, IEC61508, and takes into account not only evidence regarding requirements and design but also with the wide range of concepts related to the management of the development process in safety-critical systems.

Huhn and Hungar [7] discuss the proliferation of UML in the model-based development of safety-critical software and mention the profiles discussed above. They propose a development process where models form an integral part of the development of a safety-critical system. However, they do concede that the use of models for the certification aspect has not been adequately addressed. Our profile is a starting point for addressing this gap.

Recently, the OMG has put forward a new proposal, called the Software Assurance Evidence Metamodel (SAEM) [22], for managing safety assurance evidence. The SAEM is a standard-independent metamodel and directed towards linking the certification evidence to safety claims and the evaluation of these claims subject to the evidence. The approach that we propose uses a UML profile for characterizing the evidence of a specific standard. To perform the same task, the SAEM model will still require a definition of the specific evidence needed by a particular standard (perhaps based on a conceptual model was we have proposed). On the other hand, a profile of the SAEM could be incorporated into our approach and cover both the evidence requirements for compliance to IEC61508, as well as the evaluation of the evidence to ensure that it is sufficient to substantiate the safety claims. Together this could be a means to further the field of model-based certification.

Finally, we have used UML profiles of safety related standards in prior work [23], where we ensure that a generic standard can be specialized for a particular domain in a systematic manner. In contrast to this current paper, profiles were used in [23] as a way to keep track of the relationships between a generic standard and a sector-specific one.

7 Conclusion and Future Work

In this paper we showed how to use model-driven engineering principles and technology to specify and analyze safety evidence in order to show conformance to a safety standard. We start by establishing a sound relationship between a domain model of a safety-critical application and the evidence model of a certification standard. We do this by capturing the relevant standard as a conceptual model using a UML class diagram and using this as a basis for creating a UML profile. The profile is augmented with constraints to aid system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. The profile is then applied to a domain model of a safety-critical application aiding system suppliers in clearly demonstrating how the development artifacts of their system fulfill the compliance requirements of a standard. Constraints can be automatically checked to ensure full compliance. We illustrate our approach by presenting an excerpt of a case study that we are conducting to show how the concepts in the domain of sub-sea control systems can be related to the evidence requirements in the IEC61508 standard. We have chosen IEC61508 as the standard for illustration as it is a generic standard that applies to multiple domains and a successful application of our approach to it is a good indication of the usefulness of our work.

In future work we plan to complete the case study and assess the cost-effectiveness of our approach in the context of certification. This would mean, creating a full instantiation of a system with all the certification information included. This would allow us to compare our approach with the current industry practice of preparing for certification. We intend to check how complete our set of certification information is - how many details do we include when using our approach that are missed with current practice and how efficient is our approach

in collating all the required information compared with current practice. We also plan to extend our approach for certification of a system to multiple inter-related standards. Finally, we would like to extend our work to include the evaluation of evidence as proposed by the SAEM.

Bibliography

[1] *OMG Object Constraint Language*.

[2] Y. BAI AND Q. BAI, *Subsea Engineering Handbook*, Elsevier, 2010.

[3] K. BERKENKÖTTER, *Ocl-based validation of a railway domain profile*, in MoDELS Workshops, 2006, pp. 159–168.

[4] K. BERKENKÖTTER AND U. HANNEMANN, *Modeling the railway control domain rigorously with a uml 2.0 profile*, in SAFECOMP, 2006, pp. 398–411.

[5] P. BISHOP AND R. BLOOMFIELD, *A methodology for safety case development*, in Safety-Critical Systems Symposium, Springer, 1998.

[6] T. COCKRAM AND B. LOCKWOOD, *Electronic safety cases: Challenges and opportunities*, in Current Issues in Safety-Critical Systems, in proceedings of Safety Critical Systems Symposium, Springer, 2003.

[7] M. HUHN AND H. HUNGAR, *Uml for software safety and certification*, in Model-Based Engineering of Embedded Real-Time Systems, H. Giese, G. Karsai, E. Lee, B. Rumpe, and B. SchÃd'tz, eds., vol. 6100 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2011, pp. 201–237.

[8] *IBM Rational Software Architect*. `http://www.ibm.com/developerworks/rational/products/rsa/`.

[9] *Railway Applications âĂŞ Safety-related electronic railway control and protection systems.*, 1999.

[10] *Functional safety - safety instrumented systems for the process industry sector (IEC 61511).*, 2003.

[11] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[12] *Petroleum and natural gas industries - design and operation of subsea production systems (ISO 13628)*, 2005.

[13] D. JACKSON, M. THOMAS, AND L. MILLETT, *Software for Dependable Systems: Sufficient Evidence?*, The National Academies Press, 2007.

[14] T. P. KELLY, *Arguing Safety – A Systematic Approach to Managing Safety Cases*, PhD thesis, University of York, 1998.

[15] C. LARMAN, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)*, Prentice Hall, Oct. 2004.

[16] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[17] *Defence standard 00-55, requirements of safety related software in defence equipment (DS 00-55).*, 1997.

[18] *Safety and automation system (SAS)*, 2001.

[19] *UML 2.0 Superstructure Specification*, August 2005.

[20] *UML profile for modeling quality of service and fault tolerance characteristics and mechanisms specification*. http://www.omg.org/spec/QFTP/1.1/, 2008.

[21] *UML profile for modeling and analysis of real-time and embedded systems (marte)*. http://www.omg.org/spec/MARTE/1.0/, 2009.

[22] *Software Assurance Evidence Metamodel (SAEM)*. http://www.omg.org/spec/SAEM/, 2010.

[23] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *Using uml profiles for sector-specific tailoring of safety evidence information*, in Proceedings of the 30th international conference on Conceptual modeling, ER'11, Springer-Verlag, 2011, pp. 362–378.

[24] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[25] K. POHL, G. BÖCKLE, AND F. VAN DER LINDEN, *Software product line engineering - foundations, principles, and techniques*, Springer, 2005.

[26] *DO-178B: Software considerations in airborne systems and equipment certification*, 1982.

[27] *Application of IEC61508 and IEC61511 in the Norwegian Petroleum Industry*, 2004.

[28] G. ZOUGHBI, L. BRIAND, AND Y. LABICHE, *Modeling safety and airworthiness (RTCA DO-178B) information: conceptual model and uml profile*, Software and Systems Modeling, (2010), pp. 1–31.

Paper IV:
Planning for Safety Evidence Collection: A Tool-Supported Approach Based on Modeling of Standards Compliance Information

Planning for Safety Evidence Collection: A Tool-Supported Approach Based on Modeling of Standards Compliance Information

Davide Falessi[1,4], Mehrdad Sabetzadeh[1], Lionel Briand[5], Emanuele Turella[4], Thierry Coq[3], Rajwinder Panesar-Walawege[1,2]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

[3] Det Norske Veritas,
Paris France

[4] University of Rome Tor Vergata,
Italy

[5] Centre for Security, Reliability and Trust,
University of Luxembourg, Luxembourg

Abstract:

Safety-critical software-dependent systems such as those found in the avionics, automotive, maritime, and energy domains often need to be certified based on one or more safety standards. An important prerequisite for demonstrating compliance to software safety standards such as IEC 61508 is the collection of safety evidence. Without an upfront agreement between the system supplier and the certifier about the evidence that needs to be collected, there will invariably be important omissions, which will need to be remedied after the fact and at significant costs.

In this article, we present a flexible approach and a supporting tool for assisting suppliers and certifiers in developing an agreement about the evidence necessary to demonstrate compliance to a safety standard. The approach is model-based; specifically, the safety standard of interest is expressed via an information model. The supporting tool, which is available online, takes this information model as input and helps system suppliers and the certifiers in reaching a documented and consistent agreement about the safety evidence that needs to be collected.

1 Introduction

Safety-critical systems that depend on software – such as those found in the avionics, automotive, maritime, and energy domains – usually undergo a stringent certification process to show compliance with one or more safety standards. Although the standards provide some guidance for collecting relevant safety information for this process, the guidance is mostly textual, imprecise, and hard to specialize for context-specific needs.

An agreement about the evidence necessary to demonstrate compliance with the applicable standards is an important aspect of safety assessment practice [5]. Without such an agreement, discrepancies between the ways suppliers and certifiers interpret the standards can give rise to problems on both sides. On the supplier side, the development process might not record the information specifically necessary for certification. Recovering this information after the fact can lead to significant cost overruns and deployment delays. Indeed, given the time lapse between development and certification processes, the original developers might have moved on to a different project, department, or company. Consequently, the supplier might need to reproduce the necessary evidence from scratch, often at extremely high costs. A high-profile example of such problems occurred during the certification of the Airbus A400M computer system, when a misunderstanding in certification requirements led to substantial rework [1].

On the certifier side, problems show up when supplier documentation lacks structure and direct mention of safety information. Indeed, from our experience, suppliers often provide large fragments of their existing documents with the hope that the certifier will find the required safety information in them. The certifier must then spend the time and effort to sift through the documents and, in many cases, still not find the right information.

We developed an approach and supporting tool to systematically negotiate a consistent agreement between suppliers and certifiers about the information to collect prior to safety certification. Our approach and tool are standard-independent. However, for clarity in this article, we ground our discussions on IEC61508, a widely adopted generic standard for managing the functional safety of software-dependent systems [4].

2 A Questionnaire-Based Agreement Process

Our solution for safety-evidence planning involves a questionnaire-based agreement process, depicted in the center of Figure 1.

The process takes a safety standard's conceptual model as its input and uses a questionnaire to define details about what evidence to collect and the alternatives ways of recording and structuring it (see the Related Work in Compliance Management sidebar for more information). An administrator defines the questionnaire for a given safety standard. The supplier proposes answers (such as possible specializations), and a certifier accepts or rejects the answers, providing the decision rationale via comments.

After the certifier agrees on the supplier's answers, the tool provides as output an agreement document (in a PDF) to review, print, and sign.

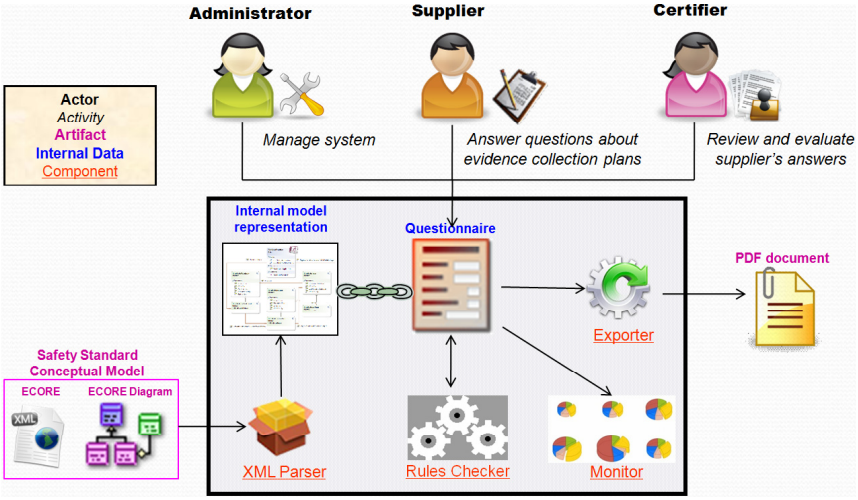


Figure 1: Overview of the solution for safety-evidence planning. An information model is encoded in an Eclipse-compatible format (Ecore) and provides the input to a questionnaire-based agreement process. Administrators, suppliers, and certifiers interact through the process to deliver a PDF document of the agreed-upon information requirements.

2.1 Model-Based Agreement

Our approach to planning safety-evidence collection is model-based. Specifically, to manage the apparent complexity of safety standards and provide an explicit and precise interpretation of the content, we use an information model that captures a given safety standardâĂŹs core concepts and their relations.

In earlier work [7], we developed an information model for the IEC61508 standard as a UML class model encoded in an Eclipse-compatible format (Ecore). Figure 2 shows a fragment of the Ecore information model. Briefly, an agreement concerning this fragment must specify which safety validation techniques are carried out in which phases and by which agents in relation to the targeted safety integrity levels. We use this model fragment later to illustrate how to build a questionnaire around the concepts and relations in an information model.

2.2 The EvidenceAgreement Tool

We developed the EvidenceAgreement tool to support our approach. The tool is web-based and lets certifiers and suppliers collaborate easily even when theyâĂŹre located at different geographical sites. EvidenceAgreement, its documentation, and its commented demo are publicly available at http://modelme.simula.no/evagr/index.html.

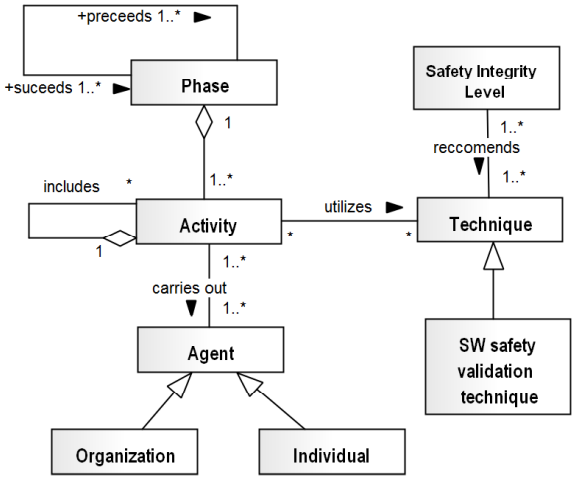


Figure 2: A fragment of the IEC 61508 information model. The main concepts of the safety standard and their relationships are formally described in a UML class diagram. The information model acts as the foundation upon which the questionnaire is created.

3 The Questionnaire

An administrator is in charge of creating a questionnaire that captures the information required to comply with a safety standard. In practice, the administrator role is typically played by one or several experts – usually, certifiers – who can interpret the relevant standard's details and enumerate alternative ways of achieving compliance in different contexts.

The administrator assigns a specific questionnaire for suppliers and certifiers to use in reaching an agreement. The supplier and certifier must subsequently authenticate themselves and then choose the questionnaire to work on among the assigned ones. The EvidenceAgreement tool lets both the supplier and the certifier monitor the questionnaire's status. As Figure 3 shows, the status for each question in both text and color-coding. Pie charts show the status for different question types in the information model and for their aggregated "Final status".

In general, each standard has one information model, but the model can include several questionnaires. As a good practice, we recommend using a single questionnaire per information model, encompassing all the domains to which the underlying standard applies. Our experience with IEC61508 supports this recommendation. However, we can't be sure that one questionnaire could support any given standard in all possible domains, so the EvidenceAgreement tool allows the association of multiple questionnaires to an information model.

An administrator can assign a pre-existing questionnaire to the supplier and certifier or create a new one. A new questionnaire should provide some basic information including the questionnaire name, author, and brief description as well as an information model based on the standard for which compliance is required. The EvidenceAgreement tool can accept any information model that can be encoded in the Ecore format.

Finally, the questionnaire must define the questions, answers, and rules as we describe in

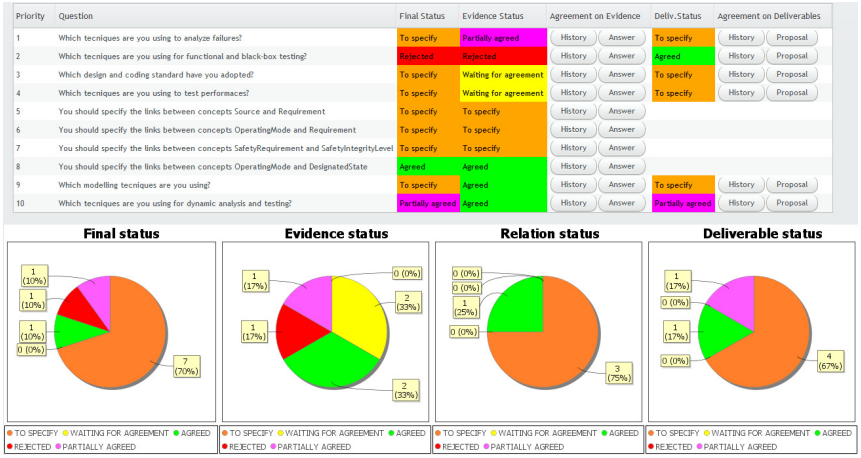


Figure 3: The EvidenceAgreement tool interface for monitoring agreement-completion status. On the one side, the supplier can easily identify which questions need an answer – namely, those that haven't yet been answered or those to which the supplier rejects the answer). On the other side, the certifier can easily identify which of the supplier's answers still require a revision.

the remainder of this article.

3.1 Question Types

The questionnaire includes five types of questions.

Context Contextual questions help suppliers better plan for evidence collection in a given context. For example, "In which domain will the product be deployed?" is a common question, and its answer would affect the safety level required. Consider a fire monitoring and control system deployed in an offshore oil platform as opposed to an on-land refinery. Each deployment would have different safety concerns and might need to comply with different safety levels.

Contextual questions are associated with the whole questionnaire, with no constraints on the number of questions or answers to each question. The supplier answers contextual questions at the beginning of the process, because the context has an overarching effect on all aspects of evidence planning. These questions are the only ones that don't require an agreement on the certifier side; the context is fixed by the supplier's obligations to its customers. For the remaining types of questions, the certifier must review and agree (or disagree) with each of the supplier's answers.

Evidence concepts Questions about evidence concepts concern the information model's classes, and the answers are textual descriptions of the types of evidence required. The administrator creates at least one question of this type for all the information model classes. For example, "Which are the adopted techniques for software safety validation?" is a question that could be associated with the "software safety validation technique" class.

Answers to these questions describe the possible specializations of the evidence. For instance, the alternative answers to the safety validation technique question in Figure 2 include "probabilistic testing", "simulation and modeling", and "functional and black-box testing".

The supplier can answer these questions by selecting from predefined answers or proposing new ones. A given questionnaire's information model stores the predefined answers. The certifier must agree on all answers to questions about evidence concepts and, if necessary, can suggest additional answers.

Relations between evidence concepts After the supplier answers the questions about the model classes, it must elaborate on the relations between the classes. The questions for a given relation are automatically derived from the answers provided for the related class pairs. Answers are of the open text type. For example, once a supplier answers the questions for "agent" and "software safety validation technique" types of evidence in Figure 2, it can specify which agent will be in charge of applying which safety validation technique.

Deliverables The certifier and supplier must agree on how to deliver the evidence. For each proposed evidence concept, the supplier must therefore answer the question, "Which deliverable(s) will provide this type of evidence?" Deliverables include artifacts (such as a given type of documentation) and actions (such as a review meeting). In the EvidenceAgreement tool, we use the Det Norske Veritas (DNV) plan-approval documentation types to populate the list of possible deliverables [2].

The supplier can choose from predefined answers or propose new ones. An agreement must have at least one agreed deliverable per evidence concept.

3.2 Rules and Inconsistent States

We use rules to enforce consistency, completeness, and traceability in the questionnaire answers. The administrator defines the rules, and the EvidenceAgreement tool checks them at runtime. There are two types of rule, which specify the constraints a questionnaire must meet:

- A *multiplicity rule* prescribes the minimum number of answers that the supplier must propose for a given question. For example, the standard may require that at least two different techniques (answers) are adopted for software safety validation.
- An *exclusion rule* excludes the coexistence of two specific answers to the same or different questions. For example, the rule excludes the answer "simulation and modeling" for software safety validation when the answer "COTS (commercial-off-the-shelf) technology" has been selected.

The interactions between exclusion and multiplicity rules can lead to inconsistent states. For example, the supplier might not be able to meet the multiplicity constraint of one question because the answers available for it are excluded by answers to other questions. To illustrate, consider the example of IEC61508, which specifies four safety integrity levels (SILs), with SIL 1 being the lowest level and SIL 4 the highest. At SIL 4, the supplier must often choose at least two testing techniques (answers) for software safety validation. If two of the three possible answers – for example, "probabilistic testing" and "simulation and modeling" –

are excluded by answers to other questions, then an inconsistency occurs because only one technique is available for software safety validation.

Inconsistent states require the user to backtrack and change the answers to one or more of the previously answered questions so that a feasible answer to the current question is no longer excluded. The EvidenceAgreement tool helps manage inconsistent states in two ways.

First, it prioritizes the questions at runtime according to the likelihood that each question will become infeasible to answer while respecting the multiplicity rules (see the first column in Figure 3). It estimates the likelihood as proportional to the number of required answers and the number of exclusion rules per answer. By following the priority order suggested by the tool, the user answers the most constrained questions first.

Second, for each alternative answer, the tool shows both all the alternatives that the answer excludes and all the alternatives that exclude the answer. The example in Figure 4(A) lists a question and three possible answers (excluded answers are hidden). The "Y/N" check-mark indicates the selected answer – in this case, "Functional and black-box testing". The blue highlight and the information in Figure 4(B) result from clicking the "Rules" button for that selection – in this case, the highlighted answer "Simulation and modeling" is excluded by an answer of the question "Do you use COTS?" In fact, the Direction column in Figure 4(B) is "in" (incoming) if the exclusion is incoming from an answer of another question and "out" (outgoing) if the selected answer excludes an answer of another question.

In Figure 4(C), the tool reports the agreement according to the date, user, role, and the specific action performed regarding the question.

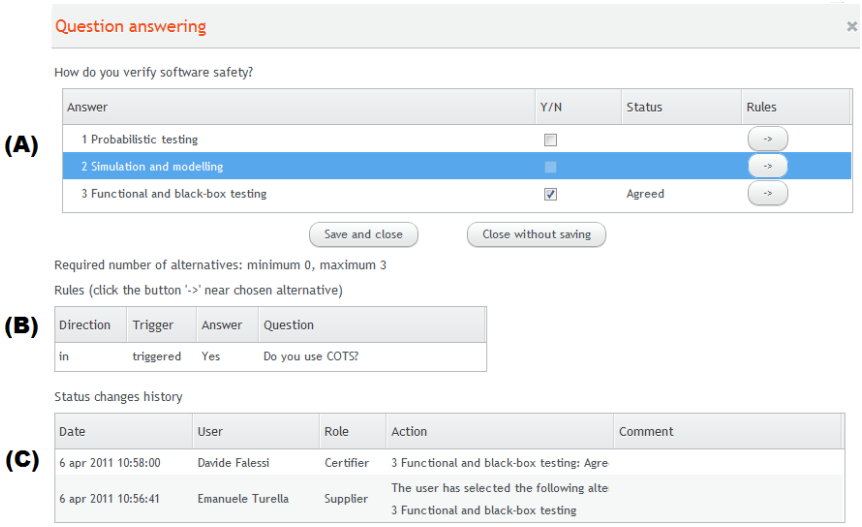


Figure 4: Example of exclusion rule type. For a given question ("How do you verify software safety?"), the tool shows (A) the possible answers, (B) the rules related to a selected answer, and (C) a log report of the agreement reached. The rules related to the answer selected in (A) are listed one per line, with the Direction column indicating whether the exclusion is incoming from, or outgoing to, the answer of another question. In this example, the highlighted answer "Simulation and modeling" in (A) has one incoming rule (from an answer of the question "Do you use COTS?") that excludes it from being a possible answer of "How do you verify software safety?"

4 Output

The tool generates a PDF document as output. This customizable report is intended mainly as an appendix to the certification contract. Figure 5 shows example output including statistical pie charts of the evidence and deliverable statuses, a matrix mapping the evidence to deliverables, and a textual description of the questionnaire results.

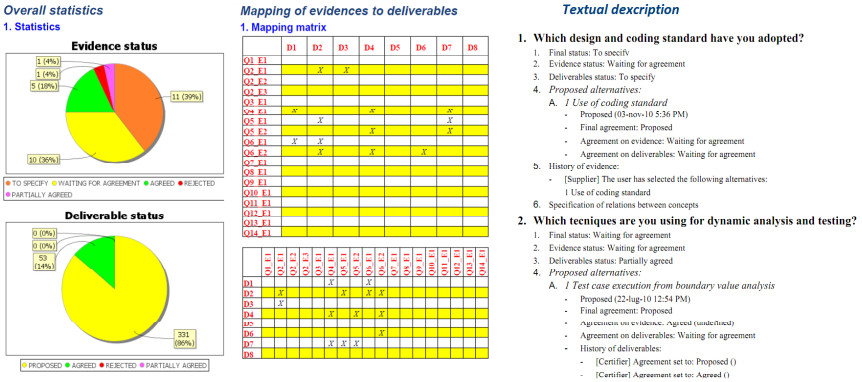


Figure 5: An excerpt of a PDF document produced by EvidenceAgreement. Both supplier and certifier can sign a printed document describing which evidence to provide; this is the result of the agreement procedure.

5 Related work

Safety certification is one facet of the more general problem of compliance management [8], which encompasses topics such as process, medical, and environment regulations. A wide array of techniques and commercial tools exist to enable the execution and monitoring of compliance-related activities. Service-level management is a related notion [9], aimed at developing a formal agreement for rendering services and ensuring their delivery accordingly.

Our work on safety certification could serve as an input to the existing compliance and service-level management tools, such as IBS's CompliantPro2 (www.ibs-us.com/en/-products/compliantpro) and MetricStream's Compliance management software (www.metricstream.com), which focus on the concrete collection and validation of evidence. In this context, our work's main contribution is the use of information models for formalizing the interpretation of safety standards and guiding decisions about what evidence to collect.

The research literature includes references to more systematic safety evidence collection as an important problem. In particular, Robert Lewis highlights the need for having a structured web of safety information covering not only hazards and safety requirements but also, among others, the requirements of development processes, hardware elements, human agents, and verification and validation results [6]. Compliance assessment schemes such as CASS [10] for IEC61508 partially address this problem by establishing guidelines for recording conformity. However, these schemes exist at a high abstraction level and must be specialized for a given domain or system. Our approach addresses this gap by helping with the specialization of safety information according to the needs of a particular context.

Our work relates most closely with questionnaire-based elicitation techniques [3]. What differentiates our work is the use of model-driven engineering concepts to facilitate the specification of questions and possible answers and thereby ensure coverage of the underlying safety standards and consistency between the provided answers.

6 Conclusion

Our approach to constructing and specializing information models for safety standards lets certifiers and suppliers develop a negotiated and structured agreement about the evidence necessary for compliance. Although we haven't formally evaluated the tool, we note that it's not a major (and disruptive) break from current practice but, instead, a more effective way to do what's already being done by manually using a plethora of different check-lists and spreadsheets.

In the future, we plan to derive data schemas from the agreements generated by our approach and use them to construct and manage safety case databases that can be analysed automatically.

7 Acknowledgement

This work was conducted as part of the ModelME! Project, which is a joint research effort between Det Norske Veritas (DNV) and Simula Research Laboratory.

Bibliography

[1] *Airbus a400m.* http://www.defensenews.com/story.php?i$=$4078604, 2009.

[2] *Recommended practice dnv-rp-a201 - plan approval documentation types*. http://exchange.dnv.com/publishing/Codes/download.asp?url=2010-04/rp-a201.pdf, 2010.

[3] W. FODDY, *Constructing questions for interviews and questionnaires*, Cambridge University Press, 1994.

[4] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[5] T. P. KELLY, *Arguing Safety – A Systematic Approach to Managing Safety Cases*, PhD thesis, University of York, 1998.

[6] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[7] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[8] M. SILVERMAN, *Compliance Management for Public, Private, or Non-Profit Organizations*, McGraw-Hill, 2008.

[9] R. STURM AND W. MORRIS, *Foundations of Service Level Management*, Sams, 2000.

[10] *Accredited certification for safety systems to iec 61508 and related standard.* http://www.61508.org/cass.htm.

Paper V:
CRESCO: Construction of Evidence Repositories for Managing Standards Compliance

CRESCO: Construction of Evidence Repositories for Managing Standards Compliance

Rajwinder Kaur Panesar-Walawege[1,2], Torbjørn Skyberg Knutsen[2], Mehrdad Sabetzadeh[1], Lionel Briand[1,2]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

Abstract:

We describe CRESCO, a tool for **C**onstruction of Evidence **RE**positories for Managing **S**tandards **CO**mpliance. CRESCO draws on Model Driven Engineering (MDE) technologies to generate a database repository schema from the evidence requirements of a given standard, expressed as a UML class diagram. CRESCO in addition generates a web-based user interface for building and manipulating evidence repositories based on the schema. CRESCO is targeted primarily at addressing the tool infrastructure needs for supporting the collection and management of safety evidence data. A systematic treatment of evidence information is a key prerequisite for demonstration of compliance to safety standards, such as IEC 61508, during the safety certification process.

1 Introduction

Safety critical systems are typically subject to safety certification based on recognized safety standards as a way to ensure that these systems do not pose undue risks to people, property, or the environment. A key prerequisite for demonstrating compliance to safety standards is collecting structured evidence in support of safety claims. Standards are often written in natural language and are open to subjective interpretation. This makes it important to develop a precise and explicit interpretation of the evidence requirements of a given standard. In previous work [4, 5], we have proposed conceptual modeling for formalizing the evidence requirements of safety standards. This approach on the one hand helps develop a shared understanding of the standards and on the other hand, provides a basis for the automation of various evidence collection and management tasks.

In this paper, we describe CRESCO, a flexible tool infrastructure for creating repositories to store, query, and manipulate standards compliance evidence. Additionally, CRESCO generates a web-based user interface for interacting with these repositories. Our work was prompted by an observed need during our collaboration with companies requiring IEC 61508 compliance. In particular, we observed that little infrastructure support has been developed to date to support management of safety evidence based on a specific standard. This issue has also been noted in the literature as an important gap in the safety certification process [2, 6]. While CRESCO is general and can be used in conjunction with different standards, we ground our discussion in this paper on IEC 61508, which is a key standard for safety certification of programmable electronic systems.

In the rest of this paper, we will describe the key components of CRESCO. For the actual demonstration, we will follow, and expand where necessary, our presentation in this paper. Specifically, the demo will begin with motivational material – similar to this paper's introduction and augmented with snippets of the conceptual model in [5]. We then go on to describe the overall architecture of the tool, as shown in Figure 1. In the next step, we will use a combination of pre-recorded and live demonstrations to illustrate the main functions of the tool, discussed in Sections 2. Finally, as we outline in Section 3, our demonstration will provide information about the tool's implementation based on our existing documentation [1], and give details about availability.

2 Tool Overview

Users can interact with CRESCO in two roles: the administrator and general user. The administrator is responsible for importing the conceptual model into CRESCO, running the transformations and setting up and starting the web server. Once the server is started, the general users – typically experts from the supplier company or certification body – can add, view and manipulate the data in the database. In this section we provide an overview of the main components of CRESCO as shown in Figure 1(a).

2.1 Generation of database schema and UI

The generation of the database and the user interface code involves two steps: a model-to-model (M2M) transformation and a model-to-text (M2T) transformation. The M2M

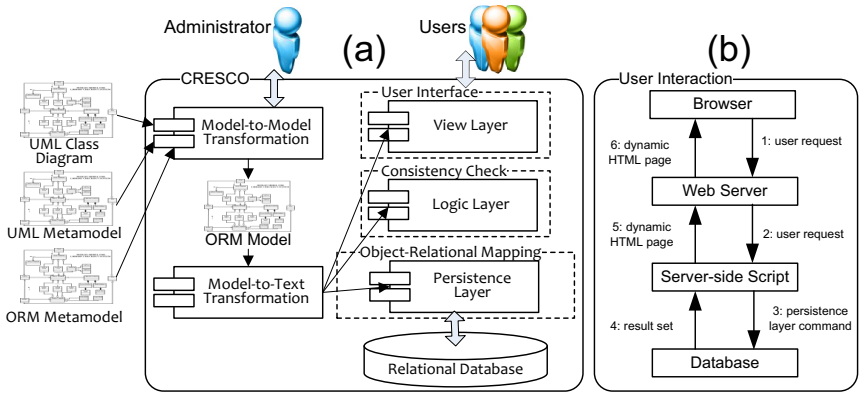


Figure 1: Components of CRESCO and How the User Interacts With It

transformation takes as input a conceptual model of a standard in the form of *a UML class diagram* [3]. This model can be created in any UML tool and then imported into CRESCO. An in-depth description of the conceptual model is beyond the scope of this demonstration paper – further details are available in [5]. The M2M transformation makes use of the UML meta-model [3] and a meta-model for an object-relational mapping (ORM) that we have created – see [1]. This ORM meta-model enables the storage (in a relational database) of objects that have been created based on a UML class diagram. The ORM meta-model includes a database schema (with tables, columns, foreign keys, etc) and object-oriented concepts, mainly generalization. The M2M transformation iterates over the conceptual model and transforms it into a model that corresponds to the ORM meta-model.

The user interface is generated from the ORM model created during the M2M transformation. The M2T transformation iterates over the elements of the ORM model and generates the database implementation as well as all the code for accessing and updating the database via a web interface. The generated code is a combination of server-side Java code and HTML (see Section 3). Figure 1(b) shows how the user interaction is processed via the generated code. Figure 2 shows the user interface generated. The left hand pane lists all the tables that have been generated and the right hand pane is used to manipulate the rows in a selected table. The 'New' button shown is used to add a new row into the selected table. Figure 2 shows the table for the concept of `Agent`, who is an entity that carries out an activity required during system development. An `Activity` is a unit of behavior with specific input and output `Artifacts`. Each activity utilizes certain `Techniques` to arrive at its desired output and requires certain kind of `Competence` by the agents performing it. The agent itself can be either an individual or an organization and is identified by the type of role it plays. In CRESCO, one can: (1) create instances of concepts such as `Agent`, `Activity`, `Artifact`, (2) fill out their attributes, (3) and establish the links between the concept instances. For illustration, we show in Figure 2, the addition of an agent record into the `Agent` table.

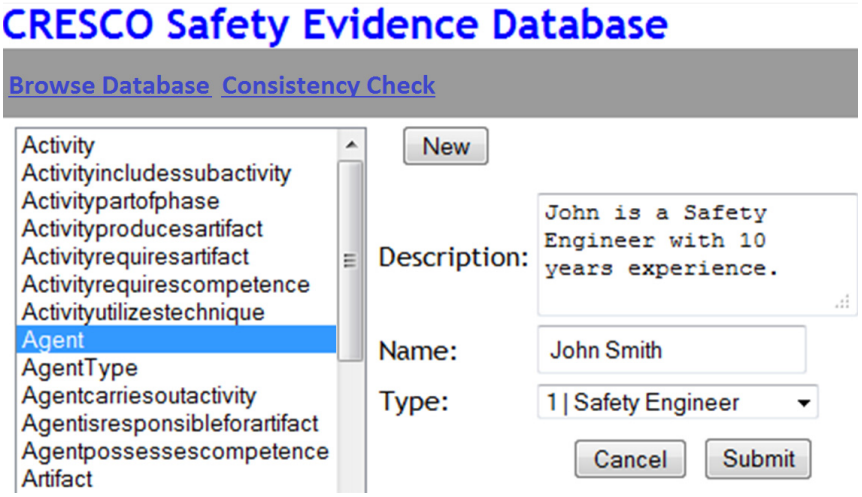


Figure 2: CRESCO User Interface

2.2 Consistency Checking

The consistency check is a means of verifying that the state of the database is in accordance with the multiplicity constraints defined in the conceptual model. The consistency check is derived from the multiplicities of UML associations. We have chosen not to preserve the multiplicities in the database schema, where all associations are represented as many-to-many. This flexibility is required so that we can tolerate inconsistencies during the construction of the database. Trying to maintain consistency at all time would be intrusive, as this would enforce an unnecessary order on how the evidence items have to be entered. While our choice allows more freedom for the user when adding entries in the database, it also calls for the implementation of a consistency checker, to verify that the data in the database is in accordance with the constraints defined in the UML class diagram. For example, an `Activity` must have at least one `Agent` who is responsible for carrying out this activity (defined in the `Agentcarriesoutactivity` table shown in Figure 2). Such constraints are checked by CRESCO's consistency checker and any violations are highlighted to the user for further investigation.

3 Implementation and Availability

CRESCO is implemented in Eclipse for Java Enterprise Edition. We use two plugins, one for Kermeta that is used for the M2M transformations and the other is MOFScript for the M2T transformations. The M2M and M2T code are approx. 800 and 1500 lines, respectively. The total number of lines of code generated depends on the size of the input conceptual model. For the IEC 61508 model, the resulting code was in excess of 20,000 lines. Hence, significant manual effort can be saved by applying CRESCO. We use Apache Derby as the underlying database which is accessed by the Java code via Hibernate. The user interface for populating

the database is via the web and we use Apache Tomcat as the web server and JavaServer Pages, Apache Struts and JavaScript to present and manipulate the objects residing in the database as well as to provide the navigation of the user interface. For our demonstration, we will present the import process of the conceptual model, the execution of the the two transformations and the user-interaction with the web-based user-interface. Due to space restrictions, we do not describe the technologies underlying CRESCO. See [1] for details and references. CRESCO is freely available at http://home.simula.no/~rpanesar/cresco/.

4 Conclusion and Future Work

We presented CRESCO a tool for the generation and manipulation of evidence repositories for demonstrating standards compliance during certification. CRESCO provides a centralized repository for keeping diverse data which, in the current state of practice, is often not collected systematically and needs to be extracted and amalgamated during certification. Our goal was to show feasibility via a coherent combination of existing open-source technologies. While our current tool provides a flexible infrastructure for managing compliance evidence, further work is required to turn it into a tool that can be deployed in a production environment. In particular, we are considering adding more sophisticated query facilities such that complex queries can be posed as well as professional reporting facilities in order to extract data from the database to create reports that can be directly given to the certification body.

Bibliography

[1] T. KNUTSEN, *Construction of information repositories for managing standards compliance evidence*, 2011. Master Thesis, University of Oslo. http://vefur.simula.no/~rpanesar/cresco/knutsen.pdf.

[2] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[3] *UML 2.0 Superstructure Specification*, August 2005.

[4] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *Using uml profiles for sector-specific tailoring of safety evidence information*, in Proceedings of the 30th international conference on Conceptual modeling, ER'11, Springer-Verlag, 2011, pp. 362–378.

[5] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[6] F. REDMILL, *Installing IEC 61508 and supporting its users – nine necessities.*, in 5th Australian Workshop on Safety Critical Systems and Software, 2000.

Paper VI:
Using Model-Driven Engineering for Managing Safety Evidence: Challenges, Vision and Experience

Using Model-Driven Engineering for Managing Safety Evidence: Challenges, Vision and Experience

Rajwinder Kaur Panesar-Walawege[1,2], **Mehrdad Sabetzadeh**[1], **Lionel Briand**[1,2]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

Abstract:

Certification is a major prerequisite for most safety-critical systems before they can be put into operation. During certification, system suppliers often have to present a coherent body of evidence demonstrating that the developed systems are safe for operation. Regardless of the certification approach taken (process-based or product-based), collection of proper evidence at the proper stage of development is critical for successful certification. Currently, system suppliers and certification bodies alike are facing various challenges in relation to safety evidence collection. Notably, they find it hard to interpret the evidence requirements imposed by the safety standards within the domain of application; little support exists for recording, querying, and reporting evidence in a structured manner; and there is a general absence of guidelines on how the collected evidence supports the safety objectives.

This paper states our position on how safety evidence should be characterized and managed. Specifically, we propose the application of Model-Driven Engineering as an enabler for performing the various tasks related to safety evidence management. We outline our current work on the specification of safety evidence requirements, upfront planning of evidence collection activities, tailoring of evidence information to domain-specific needs, and storage of evidence information. Based on this work, we identify a number of challenges that need further investigation and provide a future research agenda for managing safety evidence for software safety certification.

1 Introduction

As we create increasingly complex control systems incorporating both software and hardware, it is becoming crucial for these systems to be certified as safe for operation. A system is considered safe when it can perform its intended function without posing undue harm to the environment within which it operates. It is becoming the norm for safety critical systems to be certified by third-party certification bodies.

A key requirement of safety certification is the provision of evidence that a system complies with the applicable safety standards. The two main aspects under consideration here are: the standards involved, and the evidence that shows the compliance of the system with the specified standards. The standards prescribe the procedures for compliance and the system supplier creates, *during* the development of the system, the necessary evidence to meet the compliance requirements.

The compliance procedures may be process-based or product-based. In a process-based certification procedure, the safety of the system is assured by following prescribed activities that employ specific techniques to ensure a certain level of safety; whereas in product-based certification, a well reasoned argument that is supported by product-based evidence is required. This argumentation and evidence is usually structured into what is called a safety case [22]. Whatever the advantages or drawbacks of the two approaches, the ideal case is to have strong evidence created via a structured development process that backs the safety arguments of the product being certified. Thus we maintain that both process-based and product-based certification procedures are important for assuring the safety of a system and the commonalty between the two is the requisite compliance evidence.

In this paper, we present our experience with certification in the Maritime and Energy (M&E) industry. One of the main standards for certification in M&E is the IEC61508 standard [7] for functional safety of electrical/electronic/programmable electronic systems. Functional safety is paramount in M&E as their safety-critical systems are increasingly reliant on software. M&E now utilize various Integrated Software-Dependent Systems[1] (ISDSs) in such areas as fire and gas detection, drilling and production, vessel propulsion, steering and navigation. Hence the suppliers of these systems are increasingly required to have them certified by recognized certification bodies. In M&E, certification is important not only to assure reduction in risks but also to assure continued business. The challenge is less about whether to use process-based or product-based certification, and more about how to provide a consistent understanding of the information in the IEC61508 certification standard and how to manage the certification in a systematic and timely manner.

In this paper, we discuss the challenges with using the current certification standards in Section 2. We then propose our vision on how to tackle these challenges in Section 3. Section 4 highlights the current work we have performed in taking this vision forward. Further work that still needs to be addressed is covered in Section 5. We conclude our discussion in Section 6.

[1] Integrated systems for controlling, monitoring and maintaining safety that may be connected via communication networks.

2 Challenges in Safety Evidence Management

The use of recognized standards for certifying complex control systems is the norm for providing assurance to the public that the system will be safe during its operation. Without standards upon which to base the certification requirements, the process would become ad hoc. Standards provide a means of accumulating and sharing best practices and providing a structure to the certification process. However, the use of standards does pose some challenges to system suppliers and certifiers alike.

Creating common interpretations. Suppliers need to recognize that it is not sufficient to build safe systems, they will also need to demonstrate that a system is safe and in order to do so they need to collect the requisite evidence whilst building the system and not after the fact. Collating evidence once the system has been built can be a very expensive undertaking that may still not yield the required results. This means there should be a consistent understanding and an agreed-upon interpretation of the standard used, and all parties involved should know what evidence is to be collected and maintained in readiness for certification. This common understanding also needs to extend to the certification body – it would hardly be useful for the supplier and the certifier to have different interpretations of the standard being used. These different interpretations occur due to the standards being rather large documents expressed textually in a language not easily understood by everyone in the organization. Thus, they are amenable to subjective interpretation. This is an issue well recognized in the literature, Redmill [19] addresses it in the context of IEC61508 standard, Feldt et. al [4] discuss it regarding standards used in the space industry and most recently Sannier et. al [20] find the same problem in the nuclear energy industry. There is a need to have a common and formal interpretation of the requirements of a standard on which certification can be based.

Specializing standards to industrial contexts. A standard may need to be adapted to its context of use. It is common to have a generic standard that captures the common requirements across different sectors of industry and then derive sector-specific standards to capture the differences. We call this practice *specialization*. IEC61508 is one such standard that has been adapted to number of different sectors. In the process industry, this standard is adapted as IEC61511 [6], in railways as EN 50128 [5], in the petroleum industry as OLF070 [21], and in the automotive industry as the forthcoming ISO 26262 [8].

To effectively use derived standards, it is important to know which requirements of the generic standard map on to the sector-specific standard. This specification of the relationship between two standards can also be necessary between two standards within the same industry sector. Sometimes standards within a sector evolve, leading to different systems being specified to different versions of the same standard. In this case we again have a parent standard and another that is derived from it, and we still have the issue of systematically specifying the relationship between the two. Feldt et. al. [4] cite the lack of agreed-upon relationships between generic and derived standards as one of the main reasons behind certification delays within the space industry. Whatever the case, there has been little work to date on systematizing the specification of the relationship between generic and derived standards. Furthermore, Nordland [11] notes the lack of a well-formulated process for showing that a derived standard is consistent with a generic one. This too is directly attributable to the lack of precise and explicitly-defined relationships between the standards.

Aligning Standards to organizational practices. When a standard is being used within an

organization it will need to be aligned to the practices of the organization. In this manner, the organization can check which of its existing practices comply with the standard and which new practices need to be introduced and tailored. Thus, there is a need to assist system suppliers in relating the concepts of their application domain to the evidence requirements of the applicable standards. This observation is based on the fact that the majority of the evidence artifacts that the suppliers create and manage are based on the concepts for the application domain, as opposed to the concepts of the certification standards. The certification body also needs to interpret the standard in the context of the application domain in order to understand how the evidence relates to the standard before it can check whether sufficient evidence exists to satisfy all the requirements of the standard. This highlights the need for a systematic procedure for creating the necessary evidence and presenting it in a form that will allow the certification body to assess it in terms of both the application domain and the relevant standard.

Planning for certification. Inherent in the three challenges above is the need to ensure that the supplier and the certifier are both using the same interpretation of the standard and that both have an upfront agreement concerning the evidence artifacts that will be created during the system development. If no such agreement exists, then it is possible that the supplier may create evidence artifacts that do not match the certifiers' interpretation of the standard, or the supplier may be missing certain artifacts that the certifier deems necessary. This mismatch can be a costly affair for the supplier who will need to remedy the situation after the system has already been developed. Thus, there needs to be a systematic procedure for ensuring an upfront agreement regarding the specifics of the evidence artifacts.

Managing safety evidence electronically. The final form in which the evidence is presented for certification needs to be highly structured in order to ensure that it is readable and assessable. In general, there is a large amount of evidence that is gathered and all of it needs to be structured such that each piece of evidence and how it relates to other artifacts is clear to the certifier. Traditionally, this has been very difficult to achieve via the paper-based documents that form the basis of the certification evidence. Thus, there is a case for managing this evidence electronically [2] in order to ease navigation of the information and to allow for diversity of presentation, delivery and re-use.

Promoting re-use. The type of systems that are usually certified are characterized as belonging to product families that have many variants of a base system. Thus any proposed solution for managing certification evidence should take advantage of the re-use that is possible in these types of systems to create a systematic and cost-effective solution.

Certifying system to multiple safety standards. Control systems are often subject to certification based on multiple standards. This may occur due to the use of the system in diverse geographical locations or merely due to the diversity of the components that make up the system. The different standards that may be relevant can often have overlapping requirements and thus there is a need to effectively manage both the distinctions and the overlaps in a systematic manner.

In the remainder of the paper, we will discuss our vision for tackling the above challenges and subsequently some concrete steps we have taken to realize the vision.

3 Vision and Foundations

Having identified the challenges faced by our industry partners during certification we found there were a number of goals that any potential solution would need to fulfill:

(i) Provide ways of extracting a common understanding of the requirements presented in a textual certification standard.

  (a) Extract the most important concepts.

  (b) Extract any inherent relationships amongst these concepts.

  (c) Capture this information in a structured and systematic way such that it is can be amenable to specialization in different contexts.

(ii) Capture all requisite information electronically.

(iii) Provide some guidance for collecting safety certification information in a precise and structured manner.

Given these goals, we need a means to deals with different levels of abstraction in the information that needs to be represented: going from generic standards to specialized standards, from product family to a specific variant of a system and a way to explicitly define the relationships between the two. If we combine this with the need to structure the information systematically and electronically, we can come to the conclusion that the use of models would be an ideal way to cover all the goals.

Briefly, our position is that models, and not documents, should serve as the main sources of development information - documents, when needed, should be generated from models. For the purpose of safety certification, models are beneficial in many important respects. Most notably: (1) Models can be employed to clarify the expectations of safety standards and recommended practices, and develop concrete guidelines for system suppliers; (2) Models expressed in standard notations avoid the ambiguity and redundancy problems associated with text-based documentation; (3) Models provide an ideal vehicle for preserving traceability and the chain of evidence between hazards, requirements, design elements, implementation, and test cases; (4) Models can represent different levels of abstraction and an explicit mapping between the different levels; (5) Models present opportunities for partial or full automation of many laborious safety analysis tasks (e.g., impact analysis, completeness and consistency checking, etc).

We thus maintain that model-driven engineering techniques can be leveraged to create formalized interpretations of standards, and can serve as a primary vehicle for tackling the challenges presented in Section 2. Specifically, we have chosen UML [12] as the modelling language of choice as it is standardized and has a well-defined syntax and semantics that will give a degree of formalization to the interpretation of the standard. We use UML profiles for tailoring standards compliance evidence according to domain-specific needs. Additionally, constraints can be defined in a UML profile by using the Object Constraint Language (OCL) [1] to ensure the inclusion of compliance information in the models to which a profile is applied. We use this mechanism to provide guidance for collecting safety certification evidence. The models allow for creating electronically managed evidence that can be queried and transformed to whatever form is required by the certification body. Finally, the use of models will allow for a higher degree of re-use as we will show in Section 4.

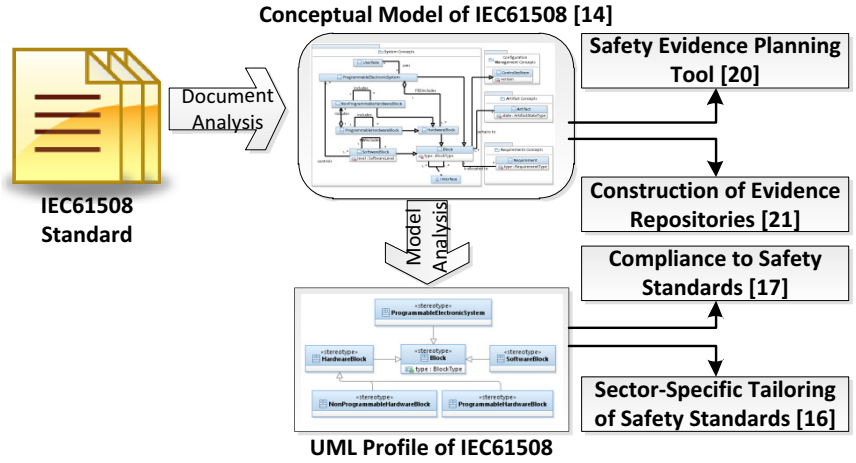


Figure 1: Our current work using MDE techniques

4 Current Work

The basis of our work is an explicit interpretation of textual standards. We need to ground our approach on a particular standard for illustration. We have chosen IEC61508 which is the most comprehensive safety standard we know of. It is a de-jour standard for many systems. It is generic and applies to multiple domains (e.g., railways, maritime, energy, process industries). Given the significance of IEC61508, we believe that the successful application of our approach to it, is a good indication of the generalizability of our work. In Figure 1, we show the areas of safety certification that we have tackled.

4.1 Creating common interpretations.

In order to create an explicit interpretation of the IEC61508 standard that promotes a common understanding of the standard, we have created a conceptual model that formalizes the evidence requirements of the standard [17]. The process of creating a conceptual model of the evidence requirements of a given standard involves a careful analysis of the text of the standard. It requires skills in modelling, systems development and knowledge of the process of certification beyond merely reading the standard. To some extent, this can be viewed as a process of qualitative data analysis, where the data is the text of the standard and it is being analyzed to identify from it, all the salient concepts and their relationships.

This retrieved information from the text is used to identify all the important concepts in the standard and as a means of explicitly showing the relationships that exist between the salient concepts. Lewis [10] expresses the need for presenting certification information as an information model. He highlights the need for creating a formal structure and the need to present the relationships that exist between atomic items of information, resulting in a *web of information* that supports certification.

In order to represent these relationships as required by a particular standard, we create a conceptual model that allows us to represent the main factors that need to be considered

for certification and the relationships amongst them [17]. The fundamental elements that we need to represent are 1) concepts, 2) attributes, 3) inter-concept relationships and 4) constraints. Additionally, as standards can be quite large, it is useful to have a means to divide the concepts into useful groupings. The conceptual model has an analogous glossary to provide descriptions of the identified concepts and their relationships. The UML [12] class diagram notation can be used to conveniently express the conceptual model. Concepts are represented as classes and concept attributes as class attributes. Relationships are represented by associations. Generalization associations are used to derive more specific concepts from abstract ones. When an attribute assumes a value from a predefined set of possible values, we use enumerations. Finally, we use the package notation to make groupings of concepts and thus better manage the model complexity.

At the most basic level, the conceptual model we have developed helps improve understanding and communication of the IEC61508 standard. Interpreting a standard like IEC61508 is a difficult task for system suppliers, and inevitably their interpretation may vary, sometimes significantly, from that of the certifier. Suppliers are frequently not clear as to what documents and information, and at what level of detail, they are expected to provide in support of safety. Furthermore, they are unsure of how these documents should be linked to hazards, requirements, activities, and so on. A concise but precise graphical representation of the core concepts in the standard such as the one we have developed is a valuable and appealing aid for understanding and using the standard. In particular, the representation can be used by the certifiers to convey their expectations and to clarify the information requirements for demonstrating compliance. Towards this end, we have used the conceptual model as the basis for creating a UML profile of the IEC61508 standard. This profile is used to *specialize* generic evidence requirements according to sector-specific needs (e.g., in the railways, avionics, and maritime and energy sectors) and to support compliance to safety standards.

4.2 Specializing standards to industrial contexts.

For sector-specific tailoring [16], we capture the relationship between the evidence requirements of a generic standard and those of a sector-specific derivation. Briefly, our approach works by building conceptual models for the evidence requirements of both the generic and sector-specific standards. The conceptual model of the generic standard is turned into a UML profile, and this profile is used for stereotyping the elements in the conceptual model of the sector-specific standard. We use OCL constraints attached to the stereotypes of the profile for validating the sector-specific conceptual model to ensure that it is consistent with the generic standard. Our approach offers two main advantages: First, it provides a systematic and explicit way to keep track of the relationships between a generic and a derived standard in terms of their evidence requirements. The concepts of the generic standard can be incorporated into the sector-specific standard whilst making a clear distinction between the two. And second, it enables the definition of consistency constraints to ensure that evidence requirements are being specialized properly in the derived standard. The consistency constraints can be automatically verified and used for providing guidance to the users about how to resolve any inconsistencies.

4.3 Aligning Standards to organizational practices

In order to support compliance to safety standards we need to establish a relationship between the concepts in the standard to the concepts in the application domain [15]. This is done by creating a domain model containing concepts that represent the physical and abstract components of a *family* (class) of systems in a particular application area (e.g., sub-sea control systems), the environment in which this family of systems function, and the key artifacts built throughout development. An example of a product family [18] is a Fire and Gas Protection system that will consist of sensors being used to detect fire or combustible gas, a controller that does processing based upon the input from the sensors and then deploys certain actuators such as sprinklers or dampers. This is a generic description of a class of systems – each variant of the system will have very specific types of sensors and actuators with specific actions that should take place upon the detection of fire or gas.

Following the norm in MDE, we assume domain models are represented as UML class diagrams [9]. This domain model is then *elaborated* using the UML profile of the relevant standard, which has been augmented with constraints to aid system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. During elaboration the stereotypes of the profile are applied to the appropriate domain model elements, and the domain model is refined so that it satisfies the OCL constraints of the stereotypes. These refinements could include the addition of new domain model elements or making changes to the existing ones (e.g., adding new attributes, revising multiplicities). Elaboration makes it possible to establish a concrete link between the evidence requirements of a given standard and a domain model. Finally, for certifying a specific system (variant) of a product family, an instantiation of the UML class diagram representing the elaborated domain model is created. In other words, an object diagram of the domain model is built to represent the specific properties of a system variant. This will represent the safety evidence to be collected to demonstrate compliance of a specific variant of the system.

4.4 Planning for certification.

The conceptual model is also used in planning for certification. Once there is an agreed upon interpretation of the standard, the certifier and supplier can use this to create an upfront plan as to what evidence the supplier will create and present for the certification process. We have created EvidenceAgreement [3], a web-based safety evidence planning tool for assisting suppliers and certifiers in developing an agreement about the evidence necessary to demonstrate compliance to a safety standard. The agreement process revolves around the notion of a questionnaire: the questions are regarding what evidence to collect and the answers are the agreed upon specifics of the evidence to collect. The tool takes the conceptual model as input and assists system suppliers and the certifiers in reaching a documented and consistent agreement about the safety evidence that needs to be collected.

4.5 Managing safety evidence electronically.

The conceptual model can be used directly to keep track of safety information by instantiating the conceptual model in a UML modeling environment. However if the suppliers do not wish to work directly with a modelling tool, e.g., due to scalability reasons, then the conceptual model

can be the basis of an automatically constructed evidence repository. We have created such a repository infrastructure, named CRESCO [14]. CRESCO is a flexible tool infrastructure for creating repositories to store, query, and manipulate standards compliance evidence. Additionally, CRESCO generates a web-based user interface for interacting with these repositories. Our work was prompted by an observed need that little infrastructure support has been developed to date to support management of safety evidence based on a specific standard. This issue has also been noted in the literature as an important gap in the safety certification process [10, 19]. CRESCO is a general tool and can be used in conjunction with different standards.

4.6 Promoting re-use

Within all this work, we have always been conscious of creating solutions that can build upon each other and incorporate a lot of re-use. In all this work, the creation of the conceptual model and the corresponding UML profile needs to be created once per standard, the domain model needs to be created once per product family and the instantiation is performed for each variant that is subject to certification. We have also chosen to illustrate our approach by working with IEC61508. Given the prolific use of IEC61508 and that our profile closely reflects its concepts makes our work reusable - the profile and its OCL constraints can be reused for all the domains that use the standard directly as well as those using its specializations. These collectively cover a significant fraction of the safety certification activities in the current practice.

Regarding the challenges of using textual standards for expressing certification requirements, we have created potential solutions and demonstrated their applicability for the first six challenges expressed in Section 2. We believe that the creation of UML profiles of the certification standards will also help in the final challenge of certification to multiple standards. We are now in the process of working on this issue. We discuss this and other future work directions in Section 5.

5 Future Research Agenda

At present, we are working on extending the process presented in [14] for the certification of a single system to multiple standards and how to deal with overlapping standard requirements. Each standard will represent different concerns, however there is likely to be some overlap in the concepts of standards that are for certifying systems in the same domain. If we choose to express each standard as a UML profile that is applied to a domain model of the system to be certified then we need to ensure that a consistent vocabulary is used such that the same terms are not used to express different concepts or the same concept is not expressed multiple times with different vocabulary. Hence, to successfully employ multiple UML profiles we will need to look for formal ways to represent the concepts in the standards such that their underlying semantics can be captured and reconciled in some automatic way.

Our current work has focused on creating models, specifically from a certification point of view. This means that we do not expect that the supplier is using model-driven development for the actual system development. The models we create are for certification, irrespective of

which development methodology is used. However, if a model-driven development approach is used for system development as well, then it should be possible to leverage those models for the purpose of certification. We would like to investigate how the conceptual model and profile of a certification standard can be used along with development models to improve the process of certification. This may have an impact on how the system is designed as the developers need to be more aware of certification requirements.

A common thread when presenting the evidence for certification is to link it to corresponding argumentation. The norm is to have safety claims and argumentation backed by evidence of how these claims are fulfilled. Recently, the OMG has put forward a proposal, called the Software Assurance Evidence Metamodel (SAEM) [13], for managing safety assurance evidence. The SAEM is a standard-independent metamodel and directed towards linking the certification evidence to safety claims and the evaluation of these claims subject to the evidence. The approach that we propose uses a UML profile for characterizing the evidence of a specific standard. To perform the same task, the SAEM model will still require a definition of the specific evidence needed by a particular standard (perhaps based on a conceptual model as we have proposed). On the other hand, a profile of the SAEM could be incorporated into our approach and cover both the evidence requirements for compliance to a particular standard, as well as the evaluation of the evidence to ensure that it is sufficient to substantiate the safety claims. Together this could be a means to further the field of model-based certification.

6 Conclusion

In this paper, we have discussed the challenges that are faced by system suppliers and certifier when having to certify systems to safety standards. These challenges are based on our experience in working in and with industry. System supplier are required to prepare for certification based on the relevant industry standards that are textually expressed and are subjectively interpreted. The suppliers run the risk of not collecting the requisite information during the development of the system and having to do so after the fact, leading to large cost overruns and delays in deployment of systems. On the other hand, certifiers may receive a large collection of documents from the supplier with the hope that the certifier will find the required safety information (based on *their* interpretation of the standard). This results in the certifier having to invest a significant amount of time and effort sifting through the provided documents, and in many cases not finding what they were looking for. What is required is a structured and systematic procedure for certification where both parties can proceed in a timely manner, being aware of what information to collect and how to navigate easily through it.

We propose that models can be used to tackle the issues that we have identified. They can be used to clarify the expectations of standards and present opportunities for automation of the certification process. To this end, we gave an overview of our current work to show the potential of using model-driven engineering techniques for safety certification. We have illustrated our work using IEC61508, one of the most commonly used standards in industry in order to show the applicability of our approach.

Bibliography

[1] *OMG Object Constraint Language*.

[2] T. COCKRAM AND B. LOCKWOOD, *Electronic safety cases: Challenges and opportunities*, in Current Issues in Safety-Critical Systems, in proceedings of Safety Critical Systems Symposium, Springer, 2003.

[3] D. FALESSI, L. BRIAND, M. SABETZADEH, E. TURELLA, T. COQ, AND R. K. PANESAR-WALAWEGE, *Planning for safety evidence collection: A tool-supported approach based on modeling of standards compliance information*, IEEE Software, 99 (2011).

[4] R. FELDT, R. TORKAR, E. AHMAD, AND B. RAZA, *Challenges with software verification and validation activities in the space industry*, in ICST'10, 2010, pp. 225–234.

[5] *EN50128 - Railway Applications - software for railway control and protection systems*, 1999.

[6] *Functional safety - safety instrumented systems for the process industry sector (IEC 61511).*, 2003.

[7] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[8] *Road vehicles – functional safety*, 2009. ISO draft standard.

[9] C. LARMAN, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)*, Prentice Hall, Oct. 2004.

[10] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[11] O. NORDLAND, *A critical look at the cenelec railway application standards*. `http://home.c2i.net/odd_nordland/~SINTEF/tekster/A_critical_look_at_rail_standards.htm`, 2003.

[12] *UML 2.0 Superstructure Specification*, August 2005.

[13] *Software Assurance Evidence Metamodel (SAEM)*. http://www.omg.org/spec/SAEM/, 2010.

[14] R. K. PANESAR-WALAWEGE, T. S. KNUTSEN, M. SABETZADEH, AND L. BRIAND, *Cresco: Construction of evidence repositories for managing standards compliance*, in Proceedings of the 30th international conference on Advances in conceptual modeling: recent developments and new directions, ER'11, Berlin, Heidelberg, 2011, Springer-Verlag, pp. 338–342.

[15] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *A model-driven engineering approach to support the verification of compliance to safety standards*, in ISSRE, 2011, pp. 30–39.

[16] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *Using uml profiles for sector-specific tailoring of safety evidence information*, in Proceedings of the 30th international conference on Conceptual modeling, ER'11, Springer-Verlag, 2011, pp. 362–378.

[17] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[18] K. POHL, G. BÖCKLE, AND F. VAN DER LINDEN, *Software product line engineering - foundations, principles, and techniques*, Springer, 2005.

[19] F. REDMILL, *Installing IEC 61508 and supporting its users – nine necessities.*, in 5th Australian Workshop on Safety Critical Systems and Software, 2000.

[20] N. SANNIER, B. BAUDRY, AND T. NGUYEN, *Formalizing standards and regulations variability in longlife projects. a challenge for model-driven engineering.*, in MoDRE workshop, 2011, pp. 225–234.

[21] *Application of IEC61508 and IEC61511 in the Norwegian Petroleum Industry*, 2004.

[22] *Defence standard 00-56, safety management requirements for defence systems (DS 00-56).*, 2004.

Paper VII:
Supporting the Verification of Compliance to Safety Standards via Model-Driven Engineering: Approach, Tool-Support and Empirical Validation

Supporting the Verification of Compliance to Safety Standards via Model-Driven Engineering: Approach, Tool-Support and Empirical Validation

Rajwinder Kaur Panesar-Walawege[1,2], **Mehrdad Sabetzadeh**[1], **Lionel Briand**[1,2,3]

[1] Simula Research Laboratory,
P. O. Box 134, N-1325 Lysaker, Norway

[2] Department of Informatics, University of Oslo,
P. O. Box 1080 Blindern, N-0316 Oslo, Norway

[3] Centre for Security, Reliability and Trust,
University of Luxembourg, Luxembourg

Abstract:

Context. Many safety-critical systems are subject to safety certification as a way of providing assurance that these systems cannot unduly harm people, property or the environment. Creating the requisite evidence for certification can be a challenging task due to the sheer size of the textual standards based on which certification is performed and the amenability of these standards to subjective interpretation.

Objective. This paper proposes a novel approach to aid suppliers in creating the evidence necessary for certification according to standards. The approach is based on Model-Driven Engineering (MDE) and addresses the challenges of using certification standards while providing assistance with compliance.

Method. Given a safety standard, a conceptual model is built that provides a succinct and explicit interpretation of the standard. This model is then used to create a UML profile that helps system suppliers in relating the concepts of the safety standard to those of the application domain, in turn enabling the suppliers to demonstrate how their system development artifacts comply with the standard.

Results. We provide a generalizable and tool-supported solution to support the verification of compliance to safety standards. Empirical validation of the work is presented via an industrial case study that shows how the concepts of a sub-sea production control system can be aligned with the evidence requirements of the IEC61508 standard. A subsequent survey examines the perceptions of practitioners about the solution.

Conclusion. The case study indicates that the supplier company where the study was performed found the approach useful in helping them prepare for certification of their software. The survey indicates that practitioners found our approach easy to understand and that they would be willing to adopt it in practice. Since the IEC61508 standard applies to multiple domains, these results suggest wider applicability and usefulness of our work.

1 Introduction

Safety-critical systems are often subject to a stringent safety certification process, aimed at providing an assurance that a system is deemed safe by a certification body. In order to structure this assurance process and the related assurance artifacts, there are certification standards that set out the requirements that system suppliers must meet. These standards can be generic standards that apply to generic technologies used across many domains or they can be very specific to a particular technology in a specific domain. There is generally overall agreement within an industry as to which standards are applicable. The suppliers of safety-critical systems are then required to present the necessary evidence to show compliance to the relevant safety standards.

Showing compliance to safety standards proves to be a very challenging task due to the fact that these standard are presented as very large textual documents that are amenable to subjective interpretation. Without an explicit interpretation of the standards, the system suppliers run the risk of missing crucial details that should have been recorded *during* system development. This would result in them having to re-construct the missing evidence at the time of certification - an often expensive and time-consuming endeavour. On the certifier's side, poorly-structured and incomplete evidence often leads to significant delays and loss of productivity, and furthermore, does not allow the certifier to develop enough trust in the system undergoing certification. It is therefore very important to devise a systematic approach, which is supported by effective automation, to specify, manage, and analyze the safety evidence necessary to demonstrate compliance to standards.

In this article, we present an approach for assisting system designers in relating the concepts of their application domain to the evidence requirements of the standards that apply to their domain. This work is motivated by a real and observed need during our work with safety-critical system suppliers. The majority of the evidence artifacts that suppliers create for certification are based on the concepts of the application domain, as opposed to the concepts of the certification standards. There is a need to link the concepts used in the application domain to those used in the standard. The absence of such a link can pose two main challenges: first, the certifier may not be able to comprehend the evidence, and second, it becomes very difficult to verify whether the evidence collected using the application domain concepts is covering all the evidence aspects mandated by the standard.

To provide a concrete example, in the IEC61508 standard, a Programmable Electronic System (PES) is the system for controlling or monitoring one or more programmable electronic devices, including all elements of the system such as sensors, communication paths, and actuators. It has software that is used to send commands for controlling the various types of equipment. A sub-sea production control system on the other hand, is made up of a Sub-sea Control Module (SCM) that incorporates a Sub-sea Electronics Module (SEM). The SCM executes the commands for opening and closing valves that control the oil well. These commands are sent from the Master Control Station (MCS) which is software that runs on the oil rig in what is called the Topside Processing Unit (TPU) [3, 16]. In this scenario, the certifier needs to know which is the PES, and which is the software system. The PES in this case is the SCM and the software controlling and monitoring it is the MCS. The correlation of these simple pieces of information provides clarification to the certifier who needs to understand the system being certified.

Solution Overview. To address the above problem, we propose a solution based on MDE for systematically guiding system designers in establishing a sound relationship between a domain model of a safety-critical application and the evidence model for a certification standard. Our proposed approach makes use of mature MDE technologies which we tailor for our specific needs. We use UML [24] class diagrams for creating domain models of the application and conceptual models of the safety standard. We then use the extension mechanisms of UML and create a UML profile of the safety standard based on its conceptual model. The profile is then augmented with verifiable constraints, written in the Object Constraint Language (OCL) [2], that help system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. In this manner, we use existing MDE technologies and tools, that are generally used for system development, and tailor them for aiding system suppliers in demonstrating how their system development artifacts comply with the requisite safety standards. In our work, we have used Rational Software Architect (RSA) [12] to provide tool support. Other modelling tools with the minimum criteria that we define in the paper in Section 4 can also be used.

Contributions. In order to empirically assess our approach, we have carried out a case study that shows the feasibility of the proposed approach and tooling in the energy and maritime domain. The results of the case study show that general modelling tools can be used in realizing our approach and that our approach provides adequate guidance to suppliers in creating the relevant evidence for certification. The case study is followed by a survey of domain experts to gauge their perceptions regarding our approach. The results of the survey show that the experts perceive benefit in adopting our approach in their certification work. Our approach uses general MDE techniques in a novel way and can be adapted to other standards. In summary, our contributions in this paper are: (1) A general approach that uses MDE techniques to aid preparation for certification; (2) the adaptation of general modelling tools used in system development to manage evidence for certification; (3) the application of the approach in the context of sub-sea production control systems; and (4) a survey of industry practitioners, presenting their perceptions of our approach.

Previously, we have studied different facets of the problem of safety evidence specification and management. Our prior work includes a conceptual framework for the specification of safety evidence using UML [30] and a technique for tailoring generic evidence requirements according to sector-specific needs (e.g., in the railways, avionics, and maritime and energy sectors) [29]. Further, the basic formulation of the approach presented in this current article has been previously published in a research paper at the 22th IEEE International Symposium on Software Reliability Engineering (ISSRE'11) [28]. This article brings together ideas described in these earlier papers and presents a definitive treatment of our approach for creating the necessary evidence to demonstrate standards' compliance. Specifically, we provide a more comprehensive description of our approach (Section 3) and tool support (Section 4), along with substantial new empirical results to show the feasibility and usefulness of our approach (Section 5).

Structure. The remainder of this paper is structured as follows: In Section 2, we discuss the challenges faced in creating certification evidence based on industry standards and the motivation for our work. In Section 3, we outline our overall approach for creating certification evidence for compliance. In Section 4, we discuss tool support, and in Section 5, we present an empirical evaluation of our approach via a case study and survey. Section 6 compares our

work with related work and Section 7 concludes the paper with a summary and suggestions for future work.

2 Background and Motivation

In this section, we briefly introduce safety certification and outline the motivations for our work.

A safety-critical system is one in which failure may lead to injury, death, or major damage to property or the environment [9]. In order to gain confidence that safety-critical systems meet their safety obligations, these systems often need to undergo certification by a certification body. The goal of certification is to provide an assurance that a system has been deemed safe for use in a specific environment. The certification process is usually based on a specific standard applicable to the domain in which the system is operated, e.g., IEC61508 [15] for the certification of electrical, electronic or programmable electronic systems that are used in safety-critical environments, IEC61511 standard for the process industry [14], EN50129 [13] for railways, and NORSOK I-002 [22] for safety automation systems in the petroleum industry.

During the certification process, the system supplier needs to provide evidence demonstrating that the safety criteria envisaged by the underlying certification standard are being met. Since the evidence that is collected for certification depends to a large extent on the relevant certification standard, there should be a consistent interpretation of the standard being used, and all parties involved (including the supplier and certifier) should know what evidence is to be collected and maintained in readiness for certification. Without an explicit and agreed-upon interpretation of the underlying standard, divergent interpretations can (and commonly do) occur because of the standards being large documents that are expressed textually and in a language not easily understood by everyone. Redmill [32] mentions these issues in the context of IEC61508, where readers have difficulty understanding the standard and engineers are unable to interpret the standard consistently throughout an organization. Feldt et. al [10] find similar challenges in the space industry, where there have been problems between customers and suppliers due to the variance that exists in the interpretation of standards. Finally, Sannier et. al [35] highlight the gaps between the possible interpretations of the same standards in the nuclear energy industry. These earlier investigations all lend support to the need for having a common and formal interpretation of the requirements of a standard upon which certification is to be performed.

Further, when a standard is being used within an organization, the practices of the organization will need to be aligned with the standard, allowing the organization to check which of its existing practices comply with the standard and which new practices need to be introduced and tailored. In order to achieve this alignment, suppliers need to relate the concepts of their application domain to the evidence requirements of the applicable standards. However, the majority of the evidence artifacts that the suppliers create and manage are based on the concepts for the application domain, and not those of the certification standards. Therefore, a systematic procedure is needed for creating the necessary evidence, such that the supplier can properly interpret the standard in the context of their application domain and verify whether sufficient evidence exists to satisfy all the requirements of the standard.

Finally, the format in which the evidence is presented for certification needs to be highly-structured in order to ensure that the evidence is readable and assessable. Traditionally, this has been very difficult to achieve via paper-based documents that form the basis of the certification evidence today. Thus, there is a case for managing this evidence electronically [7] in order to ease the navigation of the information and to allow for diversity of presentation, delivery and re-use.

The approach that we present in Section 3 uses MDE as the main vehicle for addressing the issues described above.

3 Approach

We propose an approach for assisting system suppliers with preparations for certification of their systems according to industry standards. Our solution makes use of UML profiles for specifying and automatically checking the constraints that must hold for compliance with safety standards. The solution takes into account the fact that the systems that need to be certified usually belong to a *family* (class) of systems, where each system is a variant of a base system. We consider the different levels of abstraction that are present and take advantage of the re-use that is possible in these types of systems to create a systematic and cost-effective solution.

The approach consists of four main steps as shown in Figure 1. Briefly, we start by creating a conceptual model of a certification standard (step 1). The resulting model is used for constructing a UML profile of the standard (step 2). We then apply the stereotypes of this profile to a domain model of the system that is undergoing certification (step 3). This step results in a precise link between the concepts in the certification standard and those in the system. Finally, we create instantiations of the (stereotyped) domain model for a specific certification of the system (step 4).

Steps 1 and 2 of the approach require input from experts familiar with the certification process (including the standard used for certification) but not necessarily the application domain. Fulfilling step 3 requires expertise in both certification and the application domain; whereas, step 4 only requires knowledge of the application domain. In the remainder of this section, we present detailed descriptions of the four steps in our approach.

3.1 Step 1: Conceptual Model of a Safety Standard

In Section 2, we noted the need for having an explicit interpretation of the underlying safety standard. We achieve this in the first step of our approach through the creation of a conceptual model. A conceptual model is a formal description of some aspect of the physical and social world around us for the purpose of understanding and communicating amongst humans [21]. It employs some formal notation which is a combination of diagrammatic and linguistic constructs and serves as a point of common agreement amongst a team of people and can also be used as a means of forwarding this understanding to newcomers joining the team.

A conceptual model of a safety standard should thus capture the main concepts and relationships in the evidence information required for showing compliance to the standard. We use UML class diagrams [24] for conceptual modeling of safety standards. In UML

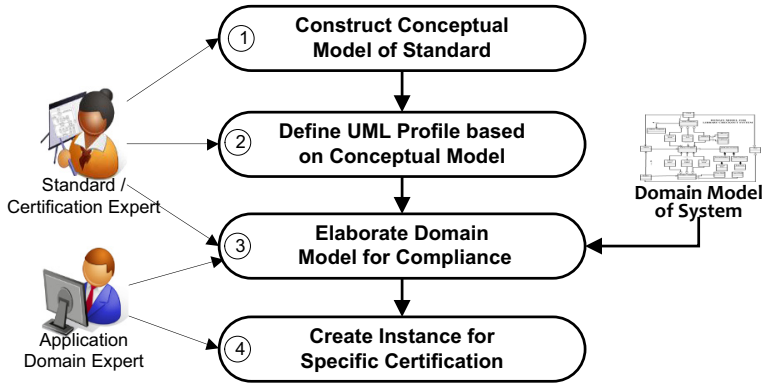


Figure 1: Methodology for the Creation of Evidence of a Safety Standard.

class diagrams, concepts are represented as classes and concept attributes as class attributes. Relationships are represented by associations. Generalization associations are used to derive more specific concepts from abstract ones. When an attribute assumes a value from a predefined set of possible values, we use enumerations. Finally, the package notation is used to make groupings of concepts and thus better manage complexity.

Our choice of UML is based on the fact that it is a well-recognized and standardized notation and that the UML class diagram notation adequately fulfils our needs. From a practical standpoint, it is in general useful to ensure that the notation being employed is already accepted in industry and at the same time easy to learn for practitioners.

Creating a conceptual model of a standard requires a careful analysis of the standard's text to identify the salient concepts and relationships mentioned in the text. To record the concepts and relationships in a systematic way, we follow a process as we read through the standard: we label each concept with a name and create a definition for it in a glossary when it is first encountered. As we proceed through the text, we either create new labels or reuse previous ones based on the definitions we have. As we create the labels, we also identify the connections between them and represent all this within a UML class diagram. This process is in line with how qualitative data analysis [8, 20] is performed in general, whereby text is analyzed to describe, classify and connect the information presented in it.

We exemplify the above process over a small excerpt of the IEC61508 standard that concerns software safety life cycle requirements. The excerpt is shown in Figure 2. In the figure, we highlight the key concepts and relationships by enclosing the relevant text in a box and numbering it.

Box 1 shows that the concepts `Phase` and `Activity` are of importance during the software development lifecycle. Box 2 identifies some important relationships between phases and activities. An activity is performed during a phase and has specified inputs and outputs. Box 3 indicates that a generic life cycle is prescribed by the standard, though deviations in terms of phases and activities are not percluded. Box 4 includes the concepts: `Technique`, `SafetyIntegrityLevel` and `TechniqueRecommendation` - indicating that activities should `Utilize` certain techniques based on the safety integrity level. The same concepts and relationships can be found in several places in the standard and thus a glossary is created to

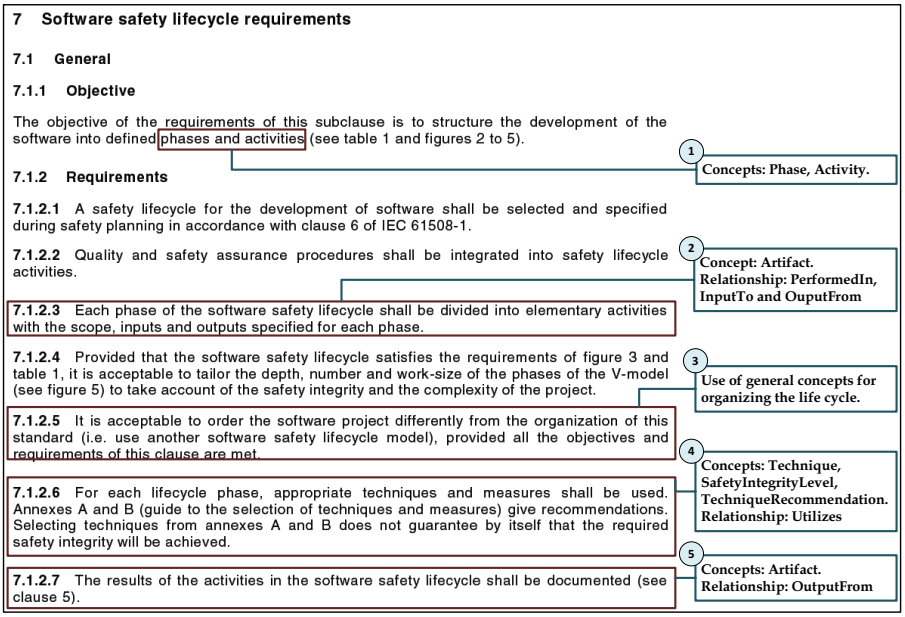


Figure 2: An Excerpt from IEC61508 showing the textual source of some of the concepts and relationships

ensure that consistent terms are used to refer to the same concepts and relationships. The glossary pertaining to the concepts shown in the excerpt can be seen in Table 1.

A graphical representation of the concepts and relationships from the excerpt is given in Figure 3. In the figure, we show some additional concepts (covered by the standard but not present in the excerpt) to aid the discussion about the process concepts in the rest of the article. We further note that the model in Figure 3 is still a partial representation of the concepts and relationships relevant to the development process. A full treatment can be found in Appendix 1.

In Figure 3, we can see that an activity can include sub activities or it can be linked to another activity by either preceding or succeeding it; these relationships are modelled by the elements `ActivityIncludes` and `ActivityLink`, along with the properties `Precedes` and `Succeeds`. Activities are to be performed by competent agents using techniques that are acceptable for the safety integrity level assigned to a component. All these aspects are modelled using the concepts `Agent`, `Competence` and `Technique` and the relationships `Requires`, `CarriesOut`, `Possesses` and `TechniqueRecommendation`. An activity may require certain artifacts as input and upon completion produce certain artifacts as output. These are modelled by the elements `Artifact`, `InputTo`, `OutputFrom`, `Requires`, `Produces`, `Input` and `Output`.

Sometimes a concept appears again, not in the same form as encountered previously, but rather as a specific case. In such instances, we use the generalization association from UML to indicate the relationship between the general and the more specific concepts. As an example, we show in Figure 4 a specific activity called `SWModuleDesignDevelopment` during which the design for software modules and their corresponding test specifications are created. It has specific input and output artifacts of different types. In Figure 4, we

Table 1: Glossary of Concepts

Concept	Description
`Activity`	A unit of behaviour in a process.
`Agent`	A person or organization that has the capability and responsibility for carrying out an activity.
`Artifact`	One of the many kinds of tangible by-products produced during the development of a system.
`Competence`	The ability to perform a specific task, action or function successfully.
`Individual`	Refers to a person.
`Issue`	A unit of work to accomplish an improvement in a system.
`Organization`	A social arrangement which pursues collective goals, which controls its own performance, and which has a boundary separating it from its environment.
`Phase`	A set of activities with determined inputs and output that are carried out at a specific time during the life of a system.
`SafetyIntegrityLevel`	The probability of a safety-related system satisfactorily performing the required safety functions under all the stated conditions within a stated period of time.
`Source`	An abstract concept that can represent a person, organization or standard that can be a source of requirements to a system.
`Technique`	A procedure used to accomplish a specific activity or task.

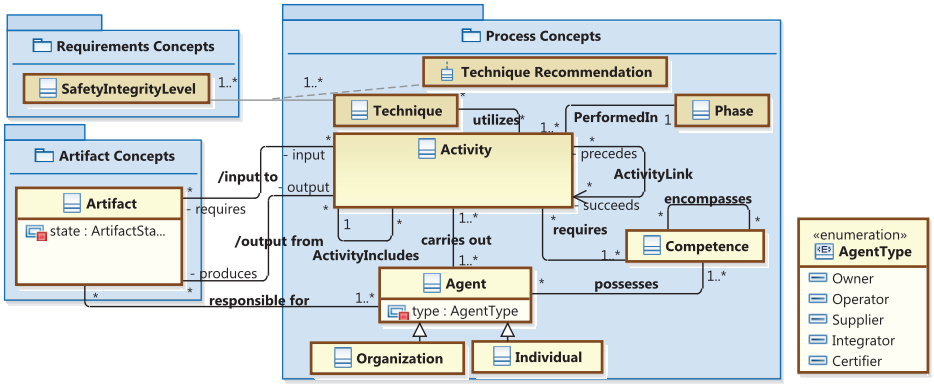


Figure 3: IEC61508 Process Concepts and their Relationships

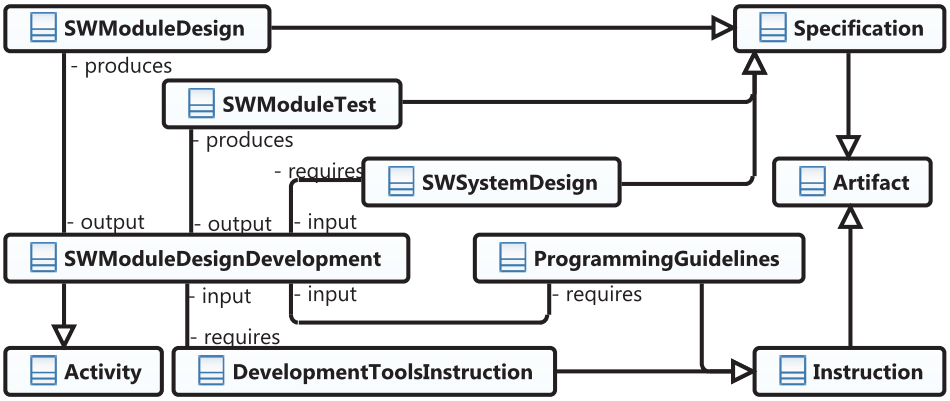


Figure 4: Software Module Design Activity

can see two types of artifacts - `Specification` and `Instruction`, which are specialization of `Artifact`. Subsequently, we have specializations of `Instruction` as `Programming-Guidelines` and `DevelopmentToolsInstruction`; and specializations of `Specification` as `SWModuleDesign`, `SWModuleTest` and `SWSystemDesign`.

The model resulting from the first step of our approach provides an explicit and precise interpretation of the evidence requirements in the underlying standard and is used in step 2 for creating a UML profile for the standard.

3.2 Step 2: Creating a UML Profile from a Conceptual Model of a Safety Standard

The conceptual model created in Step 1 forms the basis of a UML profile that we use to establish a link between the concepts in the system undergoing certification and the concepts in the certification standard being used. This link helps the supplier to verify that the collected evidence is in line with the requirements of the standard, and helps the certifier to better understand and assess the evidence artifacts provided by the supplier.

UML profiles [24] are a lightweight solution for extending the UML metamodel for use in a specific context (in our case, safety certification). They enable the expression of new concepts, notation and constraints by the introduction of context-specific stereotypes. Moreover, to ensure that certain semantics are maintained in the models to which a profile is applied, one can add constraints to the stereotypes in the profile using the Object Constraint Language (OCL) [2]. The advantage of using OCL constraints is that they can be automatically checked using an OCL validation engine, thus providing a means of efficiently ensuring that the requisite evidence items are present in the model.

We create a UML profile for a standard by having the concepts in the conceptual model of the standard represented as extensions of the metaclass 'Class' in the UML metamodel. The attributes of each concept are represented as attributes of the class to which the stereotype is applied. Relationships between the concepts are mapped as extensions of the metaclass 'Association', and the properties of an association are mapped as extensions of the metaclass 'Property'. standard- and user-specific data types are captured as Enumerations. Grouping of

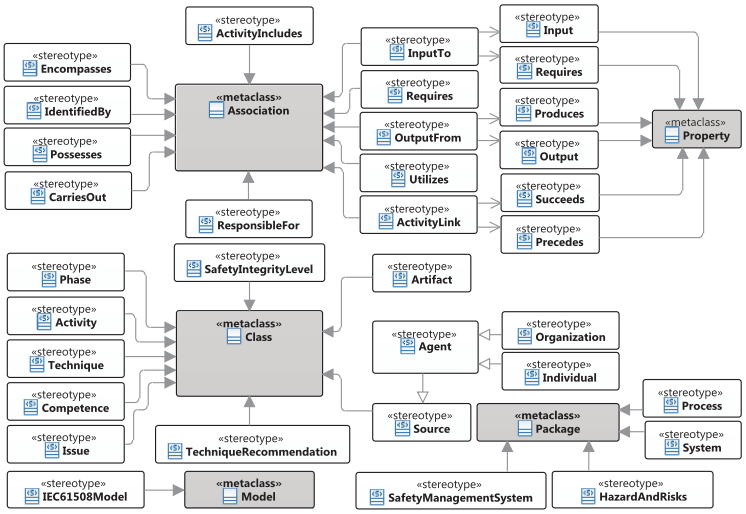


Figure 5: Fragment of IEC61508 Profile Concerning the System Development Process

model elements into packages is captured through extending the metaclass 'Package'. Finally, for a given standard, a special stereotype extending the metaclass 'Model' is defined in order to initiate the guidance process.

The actual guidance for creating the required evidence artifacts is formulated in terms of OCL constraints and attached to the stereotypes. Specifically, we use OCL constraints for the following purposes:

1: To ensure that the mandatory aspects of the standard are accounted for.

2: To ensure the application of correct stereotypes at the two ends of a given association.

3: To ensure that elements with certain stereotypes are connected to other specific elements.

4: To ensure that elements with certain stereotypes have specific properties - this helps when creating instances of the model.

5: To help with the creation of user-defined enumerations envisaged in the conceptual model.

As an example, we show in Figure 5 a fragment of the profile corresponding to the process concepts package discussed earlier along with the stereotypes that extend the metaclass 'Model' and 'Package'. We further use this fragment to show examples of the five types of constraints mentioned above.

We note that, while the OCL constraints are attached to the profile stereotypes, the constraints need to be validated on the model elements to which the stereotypes are applied. Encapsulating the constraints fully within the profile is advantageous from a usability standpoint as the user is not exposed to the constraints and only has to validate them. On the other hand, such encapsulation comes at the cost of making the constraints more complex, because to bind themselves to model elements, the constraints will have to check which stereotypes have been applied to which model elements at validation time.

The first stereotype to be applied is to the model itself, in our example using IEC61508, this is the `IEC61508Model` stereotype, which starts the incremental guidance process about which types of evidence to create. This stereotype has attached to it OCL constraints that ensure that certain packages are present in the model to organize the different model elements (constraints of type 1). As an example, we show the constraint on this stereotype for ensuring that the `HazardsAndRisks` stereotype exists on a UML package in the model[2]:

```
self.base_Model.allOwnedElements()->
  exists(e | e.oclIsTypeOf(uml::Package)
   and not e.getAppliedStereotype(
    'IEC61508Profile::HazardsAndRisks').
      oclIsUndefined())
```

The keyword `self` refers to the element being constrained, in this case the `IEC61508Model` stereotype. Properties and attributes of an element are referenced using the dot notation. The `base_Model` reference is used to access the model to which the stereotype has been applied, which would be the domain model in our case. The `allOwnedElements` is an operation that returns all the elements in the model. `exists` is an OCL operation that will check that at least one element in a collection of elements satisfies the given constraint. The constraint in the `exists` clause specifies that at least one element is of type `Package` using the operation `oclIsTypeOf` and that this element also has the stereotype `HazardsAndRisks` applied to it (using the operation `getAppliedStereotype`).

To model the development life-cycle, we have the stereotype `Phase` to model the different phases in a life-cycle and the stereotype `Activity` to model the activities. The `Phase` stereotype has a constraint attached to it that states that every phase must have at least one `Activity` defined for it, i.e., there must be elements that have the stereotype `Activity` in the same package and be connected to the element with the stereotype `Phase`. This is enforced through two different constraints, one on the class stereotype `Phase` and the other on the association stereotype `PerformedIn` (see Figure 3). On the stereotype `Phase`, we have a constraint (of type 3) that states that there should be an association with a stereotype `PerformedIn` originating from the element that has this stereotype:

```
self.base_Class.ownedAttribute->
  collect(c:Property | c.association)->
   select(a:Association | not a.getAppliedStereotype(
    'IEC61508Profile::PerformedIn').
      oclIsUndefined())->size()>0
```

On the stereotype `PerformedIn`, there is a constraint that states that this stereotype can only be applied to an association that is between a pair of elements that have the stereotypes `Activity` and `Phase`, respectively (a type 2 constraint):

```
self.base_Association.memberEnd->
  select(p:Property not (
   p.class.getAppliedStereotype(
```

[2]For the sake of brevity, we have omitted the name and context of the constraints shown

```
    'IEC61508Profile::Activity').
      oclIsUndefined()))->size()=1
and
self.base_Association.memberEnd->
  select(p:Property| not(
   p.class.getAppliedStereotype(
    'IEC61508Profile::Phase').
      oclIsUndefined()))->size()=1
```

Constraints are also used to create properties for the elements on which stereotypes have been applied, or for creating user-defined types. For example, an element with the `Artifact` stereotype applied to it should have a property called 'State' of type 'ArtifactStateType' which is a user-defined enumeration (this combines constraints of types 4 and 5):

```
self.base_Class.ownedAttribute->
  one(p:Property| p.name='State' and
   p.type.name='ArtifactStateType' and
    p.type.oclIsTypeOf(uml::Enumeration))
```

The above types of constraints allow the user to define domain-specific values for the enumerations (the profile only gives the name and type of the property).

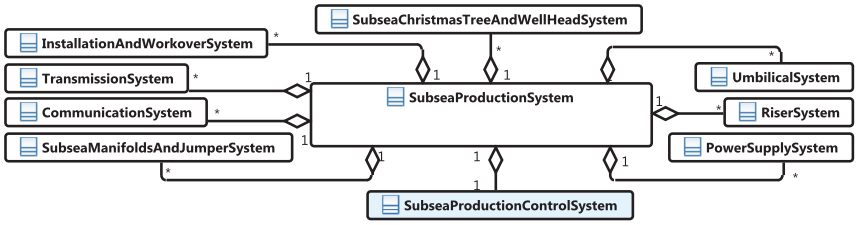
The stereotypes together with the constraints defined on them are used in Step 3 to build a precise link between the concepts in the application domain and those in the standard's domain, and to provide systematic guidance on how to create these links. We name this process *elaboration* and explain it in the next step.

3.3 Step 3: Elaborating a Domain Model for Compliance.

Once we have the UML profile created in step 2, we can proceed to apply the stereotypes of the profile to the elements of a domain model of the system to be certified. A domain model is a visual representation of real-world concepts and the relationships amongst them in a specific area of interest [18]. In the context of our work, we use the term domain model to refer to the concepts that represent the physical and abstract components of a class of systems in a particular application area (e.g., sub-sea production systems), the environment in which the system functions, and the key artifacts built throughout development.

We use sub-sea production systems to exemplify what constitutes a domain model in our context. In sub-sea production systems, there are a number of subsystems working together to extract the oil from the sea bed. In Figure 6, we show a simplified decomposition of sub-sea production systems into their constituent subsystems along with a brief description of the subsystems.

Within these large systems, we concern ourselves only with the Sub-sea Production Control System. We show a fragment of the control system and how it interacts with the other components in Figure 7. In this figure, we can see that the wellhead attaches to the CT (see the descriptions given earlier in Figure 6). The CT connects to the manifold that is anchored to the seabed via a structural frame called the template. Mounted on the CT is the Sub-sea Control Module (SCM) that receives commands from the Master Control Station (MCS) that is part of



A Christmas Tree and Well Head System has wellheads that attach to the sub-sea oil or gas wells and an assembly of control valves, pressure gauges, and chokes put on the top of a well to control the flow of oil and gas, known as a Christmas Tree (CT). The CT is housed on a manifold that is part of the Manifold and Jumper System. The Umbilical System is a housing that carries the power and communication line from the surface to the subsea equipment whereas the Riser System consists of all the equipment carrying the oil from the well to the surface. The Transmission and Communication systems are responsible for conveying the signals sent from the surface control equipment down to the sub-sea equipment. The Installation and Workover System is used to control and monitor sub-sea equipment during installation or maintenance. Finally, the Sub-sea Production Control System controls the valves and chokes on the CT and anywhere else on the manifold depending on the design of the system. This is done by sending and receiving data between the surface and the sub-sea equipment, thus allowing the engineers at the surface to monitor the sub-sea equipment.

Figure 6: Subsystems in a Sub-sea Production System

the Topside Processing Unit (TPU) located in the Sub-sea Power and Communication Unit (SPCU). The SCM also sends signals from the sub-sea instruments to the MCS. The signals are sent from the SCM via the Sub-sea Router Module (SRM) to a router in the SPCU that passes the signal to the MCS. A more complete description of these components can be found in [3, 16].

The model in Figure 7 is a generic description of a class of systems – each variant of the system will have very specific types of oil wells, manifolds and sensors and actuators with specific actions that should take place in order to extract the oil. Following the norm in MDE, we assume domain models are represented as UML class diagrams [18]. We do not concern ourselves in this article with the construction of domain models. Good references and guidelines already exist [18].

To establish a mapping between a domain model and a standard, we apply stereotypes from the UML profile of the standard to the domain model. Specifically, the elaboration of the domain model means the application of the profile stereotypes to the appropriate domain model elements, and refining the domain model so that it satisfies the OCL constraints attached to the stereotypes. These refinements could include the addition of new domain model elements or making changes to the existing ones (e.g., adding new attributes, revising multiplicities).

As we stated in Section 3.2, our approach envisages a special stereotype extending the 'Model' metaclass for starting the guidance process. This stereotype is applied to the domain model itself, specifying what standard the domain model needs to comply with. We continue our exemplification using IEC61508. Thus, the `IEC61508Model` stereotype is applied to the domain model and the OCL constraints of this stereotype are validated. The violated constraints, shown in Figure 8, are the beginning of the guidance process for creating the evidence. The first requirement is the creation of four packages that have the stereotypes: `Process`, `System`, `SafetyManagementSystem` and `HazardsAndRisks`.

The rationale behind requiring these packages comes from the IEC61508 standard. The

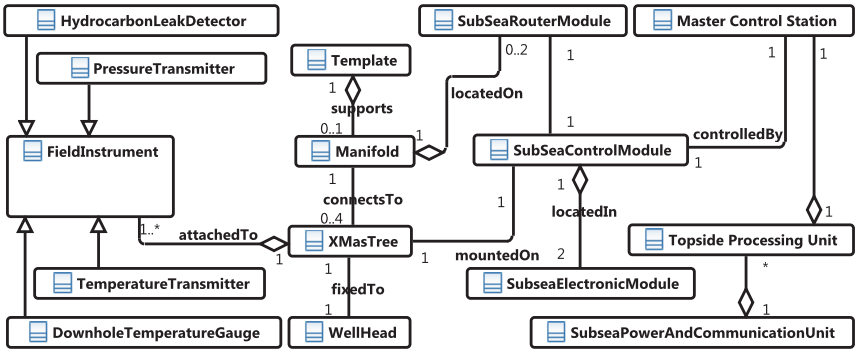


Figure 7: A Domain Model Fragment of a Sub-Sea Production Control System

IEC61508 standard promotes a risk-based approach for determining the required level of safety measures for safety-relevant systems, hence the need for the `HazardsAndRisks` package. Risks can only be determined based upon the hazards that will exist when the system is used, thus it is important to have a breakdown of the system, bearing in mind both the hardware and software aspects of the system as well as the role of human users. This breakdown will be kept in the `System` package. The standard also puts emphasis on having clearly specified technical and management activities and a clear identification of all responsible persons within the organization that perform these activities. The management information is kept in the `SafetyManagementSystem` package whereas the technical activities specified within a safety life-cycle and kept in the `Process` package.

The packages themselves can be named using the supplier's own terminology, but the specified stereotypes need to be applied. Once the package stereotypes have been applied, the next set of (violated) constraints will provide guidance on what stereotypes to apply next.

Figure 9 shows that five constraints have failed once the package stereotypes have been applied. Hazards have not yet been identified in the `HazardsAndRisks` package; phases have not yet been identified in the `Process` package; agents and their competence have not been identified in the `SafetyManagementSystem` package, and blocks have not been identified in the `System` package.

First, we will show the application of the stereotypes that identify the system components. These stereotypes are shown in Figure 10. The stereotypes describe the basic elements needed to conceptualize safety-related control systems that involve both hardware and software. A Programmable Electronic System (PES) is represented by the stereotype `Programmable-ElectronicSystem` and is made up of one or more hardware blocks represented by the stereotype `HardwareBlock` and controlled by a number of software blocks - stereotype `SoftwareBlock`. A hardware block may represent a mechanical, electrical or electronic entity, both programmable and non-programmable, hence the existence of stereotypes `Non-ProgrammableHardwareBlock` and `ProgrammableHardwareBlock`.

By applying the stereotypes relevant to the IEC61508 standard, we are now relating the elements of the system as they pertain to the concepts described in the IEC61508 standard. The software of the MCS controls and monitors the sub-sea wells, so the stereotype `Software-Block` is applied to this element. The system being controlled is the SCM which contains the

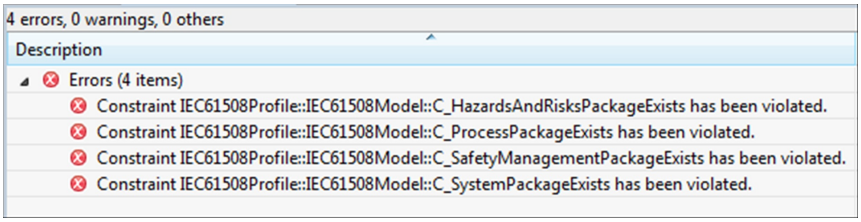


Figure 8: Error Report Showing the Violated OCL Constraints of the IEC61508Model Stereotype

Sub-sea Electronic Module (SEM), which in turn links to the different instruments. Thus, the stereotype `ProgrammableElectronicSystem` is applied to the SCM and the stereotype `ProgrammableHardwareBlock` to the rest of the elements. The applied stereotype are shown in Figure 11.

If we now focus on the software and validate the MCS element, the validation will fail. In Figure 12, we show the violated constraints. All system blocks need to have a unique identification and this has not been added to the MCS element – in the petroleum industry, every components of a system has a unique identifier called a tag. The standard also recommends version control of the system components - a version attribute has not yet been added either, thus the violation of that constraint. A constraint on the `SoftwareBlock` stereotype requires one to show the decomposition level of the software, i.e. whether it is the entire software system or are we referring to a software module that is part of a system. The constraint specifies the name of the attribute ('Level') and the type ('Enumeration').

There are also various certification artifacts that need to be created during the construction of the software. These, among others, may be plans that guide the process of software construction (e.g., software verification plan), technical guidelines for the programmers such as programming guidelines or development tool instructions, and results of testing the software to show that it meets it's requirements. All these are shown as required certification artifacts via the violation of the OCL constraints. In total, from Figure 12, we see that there are 25 different constraints that must be met for the certification of the software of the MCS.

As it would not be possible to show all the artifacts in a small legible diagram, we show in Figure 13 the resulting model after eight different artifacts have been added to satisfy some of the constraints. We have added an element to depict the software safety requirements (stereotyped with `SWSafetyRequirements`) that are created during the requirements analysis activity and a software validation plan (stereotyped with `SWSafetyValidationPlan`) which is the output of a safety validation planning activity. During the architecture design activity, the software architecture description is created (stereotyped with `SWArchitectureDescription`), as well as the software integration test specification (stereotyped with `SWArchitecture-IntegrationTest`) and how to test the software with the hardware is detailed in the software/hardware integration test specification (stereotyped with `SW_PE_IntegrationTest`). The software system design activity results in the system design specification (stereotyped with `SWSystemDesign`) and the support tool and programming language selection activity results in the tool selection (stereotyped with `DevelopmentToolsInstruction`) and coding standards (stereotyped with `ProgrammingGuidelines`) to be used for development.

The unique identification, version and software level attributes have been added as well as a

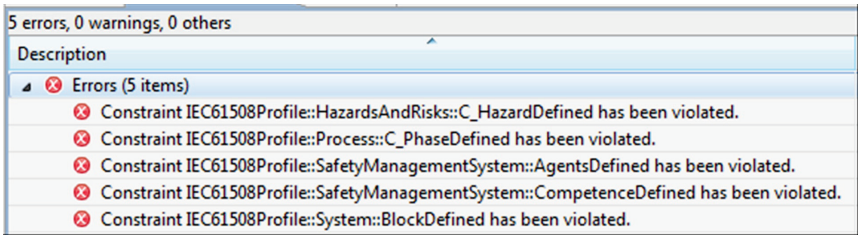


Figure 9: Error Report Showing the Violated OCL Constraints After Application of the Package Stereotypes

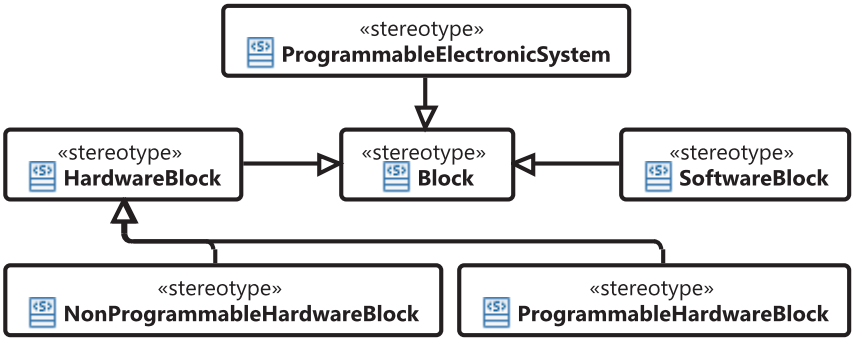


Figure 10: IEC61508 Profile fragment for System Stereotypes

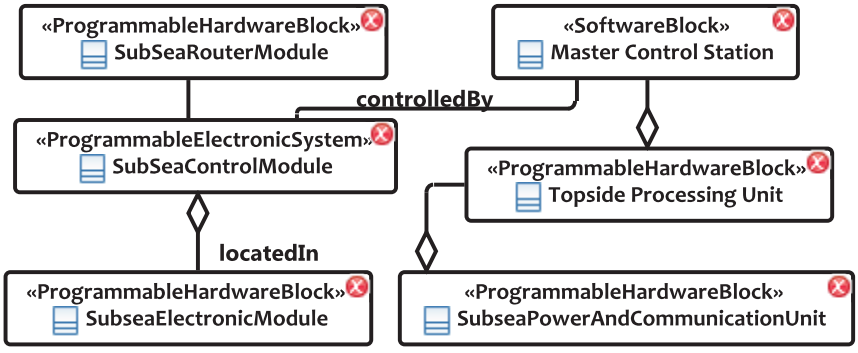


Figure 11: Fragment of the Domain Model After Application of the System Stereotypes

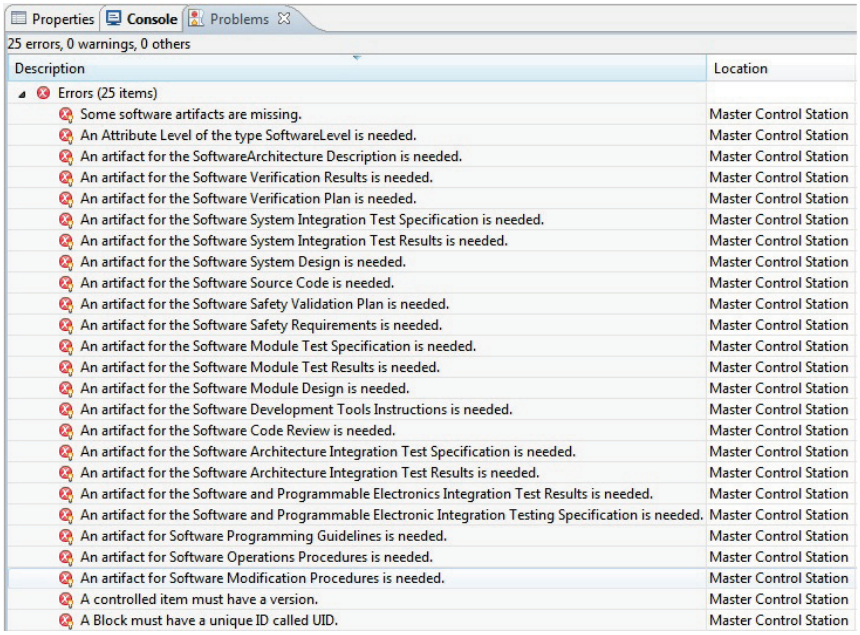


Figure 12: Error Report Showing the Violated OCL Constraints for the Master Control Station

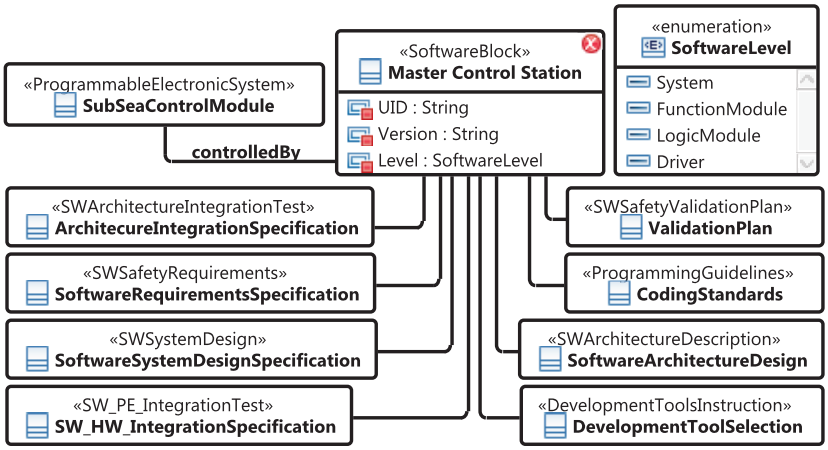
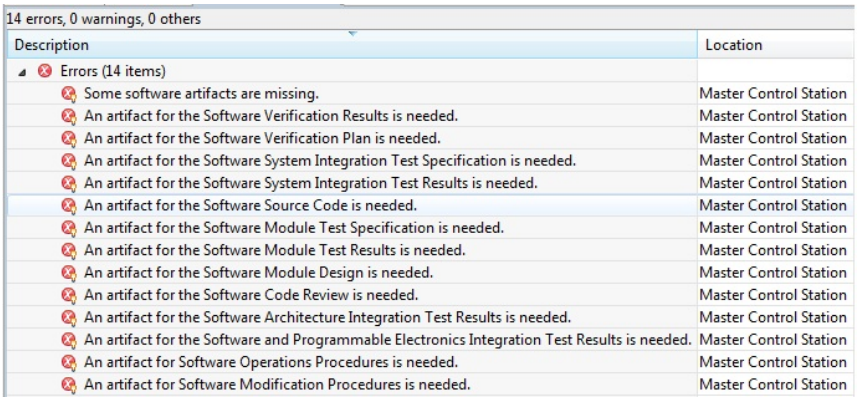


Figure 13: Fragment of the Domain Model After Stereotyping and Adding New Elements During Elaboration



14 errors, 0 warnings, 0 others

Description	Location
Errors (14 items)	
Some software artifacts are missing.	Master Control Station
An artifact for the Software Verification Results is needed.	Master Control Station
An artifact for the Software Verification Plan is needed.	Master Control Station
An artifact for the Software System Integration Test Specification is needed.	Master Control Station
An artifact for the Software System Integration Test Results is needed.	Master Control Station
An artifact for the Software Source Code is needed.	Master Control Station
An artifact for the Software Module Test Specification is needed.	Master Control Station
An artifact for the Software Module Test Results is needed.	Master Control Station
An artifact for the Software Module Design is needed.	Master Control Station
An artifact for the Software Code Review is needed.	Master Control Station
An artifact for the Software Architecture Integration Test Results is needed.	Master Control Station
An artifact for the Software and Programmable Electronics Integration Test Results is needed.	Master Control Station
An artifact for Software Operations Procedures is needed.	Master Control Station
An artifact for Software Modification Procedures is needed.	Master Control Station

Figure 14: Error Report Showing the Remaining Violated OCL Constraints for the Master Control Station

user-defined enumerated type for the 'SoftwareLevel'. The actual literal values are set by the user as relevant to their industry. In this case the literals used are 'System', 'FunctionModule', 'LogicModule' and 'Driver' - these were the values that were most relevant for our industry partner. When the MCS element is validated after the addition of the new elements to the model, we see that the constraints related to the elements we have added are not violated any more and we can work on the remaining ones. We show the remaining violated constraints in Figure 14.

The new artifact elements that were linked to the MCS element would be defined in the process package. This was one of the packages that was required at the beginning of the elaboration process. The elements in the process package can be defined as needed, or be all created prior to their need and the links to the the MCS element established when the constraints fail. In Figure 15, we show some of the elements in the process package. Specifically, in the figure, we show the artifact elements that we have used in Figure 13. The artifact elements are shown on the left in blue. In the center we have the activities that lead to the creation of these artifacts, shown in orange. The development phase is software development, shown in green. For each activity, the standard recommends that the competence required should be documented as well as the agent who carries out the activity. We show an example of this for the `SWSafetyRequirementsSpecification` activity, coloured yellow.

We can see that the user can name the elements according to their own conventions, it is the stereotypes that are specific to the standard. For example, the software safety requirements element is named by the supplier as 'SoftwareRequirementsSpecification' and has the stereotype `SWSafetyRequirements`. It was created as an output from the activity called 'RequirementsAnalysis' and the analogous activity from the standard is identified by the stereotype as `SWSafetyRequirementsSpecification`.

In the above example, we presented very simply, how elaboration happens and new elements are added or existing ones are linked to new elements in order to satisfy the OCL constraints linked to the stereotypes. We have only shown the constraints for some of the elements, but all elements have such constraints and through this process of satisfying the OCL constraints, the domain models are updated to satisfy the requirements of the applicable safety standard. For ease of explanation, in the examples above, we show the validation of constrains per element,

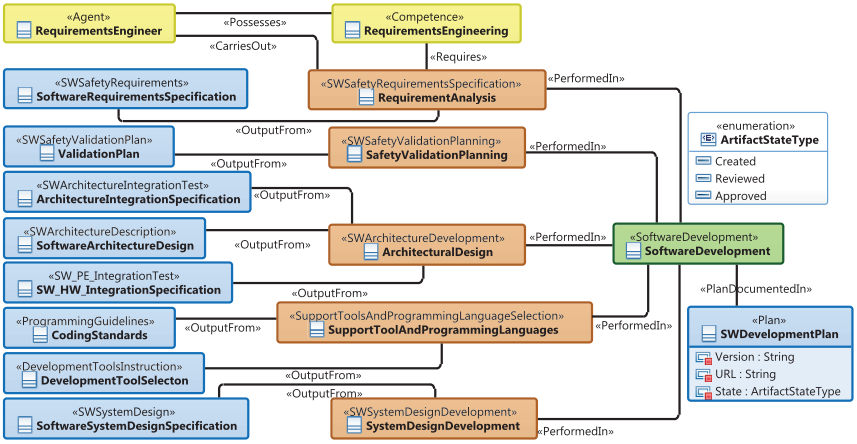


Figure 15: Fragment showing the Software Development Cycle according to IEC61508 Standard.

but they can be validated per diagram or per an entire model as well. Therefore, it is possible to work on the model in small increments while still being able to see what the overall state of the model is in regards to satisfying the requirements of the relevant standard. Once the domain model has been fully elaborated, it can be used for specific certifications of that class of systems as presented in step 4.

3.4 Step 4: Creating an Instance for a Specific Certification.

To support the certification of a specific system variant from a class of systems, the fourth and final step of the process in Figure 1 is performed. This step creates an instantiation of the UML class diagram representing the elaborated domain model. In other words, an object diagram of the elaborated domain model is built to represent the specific properties of a system variant, and instances of the certification evidence are created as specified by the elaborated model. Note that whereas steps one and two of our approach are performed once per standard, and step three once per class of systems, the fourth step is performed once for each variant that is subject to certification.

In Figure 16, we show an instance model conforming to the domain model that we presented in Figure 13. This instance model is partial in two respects: first, we do not show instantiations of all the elements in the model of Figure 13 and second, for those elements that we do instantiate, there is only one element instance. In an actual system, the number of instantiations per element can be more. For example, we show only one instance of each of the elements 'PressureTransmitter' and 'TemperatureTransmitter' whereas in an actual system there are numerous pressure and temperature transmitters. Also, for confidentiality reasons, we use sanitized names and do not give the real names of the element instances.

In the instance model shown, there is one template and one manifold. The manifold has two CT structures on it - X1 and X2, each connected to a wellhead. X1 has a temperature transmitter (TT1) attached to it and X2 has a pressure transmitter (PT1) attached to it. There are two SCMs, both controlled by a single MCS. The unique Id given to the MCS is 'T1823a',

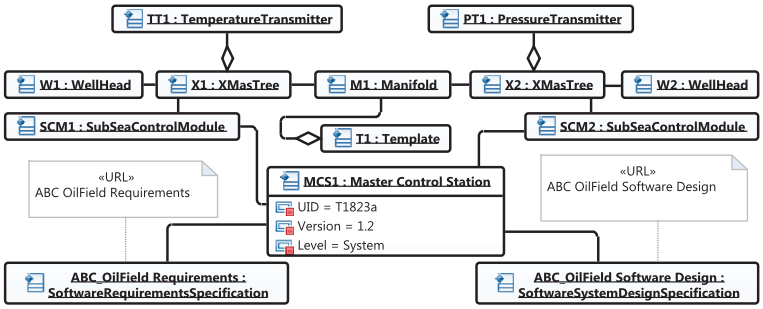


Figure 16: Instance Model Created from the Elaborated Domain Model

the version of the software for the MCS is 1.2 and this is the system level software version. These attributes were added to the MCS element due to constraints on the stereotype, allowing specific values to be set in the instance model. For the hardware equipment, the version would store the model number or serial number of the piece of equipment. An instance of the software system requirements artifact called the `ABC_OilField Requirements` is shown, as is an instance of the software system design artifact called the `ABC_OilField Software Design`. Each of these artifacts has a URL linked to it so that that actual document can be accessed right from the tool environment. In this manner, the documents can be stored in any location and are simply accessed via a URL. This provides a mechanism that links diverse artifacts in diverse locations.

In this way, the evidence requirements for a specific system can be created in readiness for certification based on the relevant standard. Further requirements in terms of tool support for applying this approach are presented in Section 4 and the results of applying this approach in Section 5.1.

4 Tool Support

The tool support for our approach has to fulfill the following key requirements:

- Allow the creation of UML class diagrams which we use as a notation for representing our conceptual model.
- Allow the creation of a custom UML profile.
- Support the creation of OCL constraints at the level of the profile.
- Support the validation of OCL constraints.
- Provide customization of the messages given to the user when a constraint is violated.
- Provide the ability to create instances of the elaborated models.
- Provide the ability to create customized reports by querying the constructed models.

We have chosen Rational Software Architect (RSA) [12] by IBM to provide tool support for our approach. In addition to meeting all the above requirements, RSA is a mature and industry-strength tool with good usability, thus making it easier to apply our approach in an industrial setting and making it more likely for the approach to be adopted by practitioners.

We have successfully used RSA version 8.03 in our case study to support all the steps of our approach described in Section 3. Specifically, we used RSA to create the UML class diagrams for the conceptual model of the standard as well as the domain models for elaboration. We then used RSA to create the UML profile of the IEC61508 standard. RSA supports adding OCL constraints at the level of the profile. More importantly, it has a built-in OCL validation engine that we could utilize to provide the guidance for elaborating the domain models according to the IEC61508 profile. The messages given to the user when a constraint is violated can be customized. RSA also includes a report designer based on Business Intelligence Reporting Tool (BIRT) [1], that can be used to publish reports in user-defined layouts based on the data in the models. While we have not yet customized this report designer for generating safety certification reports, the existence of such a flexible report generation framework was an important consideration that we had to account for.

The domain models can be created in a hierarchy. This allows one to start with a high-level view and then create more detailed models as and when necessary. Large diagrams can also be split into a number of smaller diagrams, but if an overall view of a particular element is required, then a *'browse'* diagram can be automatically generated. A 'browse' diagram shows all the elements that a chosen element is related to and helps in understanding how that element fits into the overall system depicted in the model providing a snapshot of the overall context of an element. These diagrams are not permanent diagrams: they are generated from the most current information in the models and hence a browse diagram can be refreshed to show the latest state of the model elements. It is also possible to convert a browse diagram to an editable diagram. This provides a means to both get an overall context of an element and proceed to edit it if necessary.

RSA further allows for custom documentation to be added to the stereotypes. When the mouse cursor hovers over a stereotype, a pop-up window displays the associated documentation, as shown in Figure 17. All stereotypes can be documented in this way to provide further assistance to the user while applying the stereotypes.

To help in the creation of the instance models, there is a properties view that shows the slots for the selected instance. Each slot is a mapping to an attribute of the classifier that has been instantiated and every time a value is created for a particular attribute, the properties view is updated to reflect the change. The creator of the instance model can thus see which slots have values already and which ones still need values. In this way, RSA can guide the user in creating a complete instance model.

5 Evaluation

In this section, we present an experimental validation of our approach. First, in Section 5.1, we report on an industrial case study performed in the maritime and energy domain. This is followed in Section 5.2 with the description of a survey performed among domain experts to better understand their perceptions about our approach. The case study enables us to

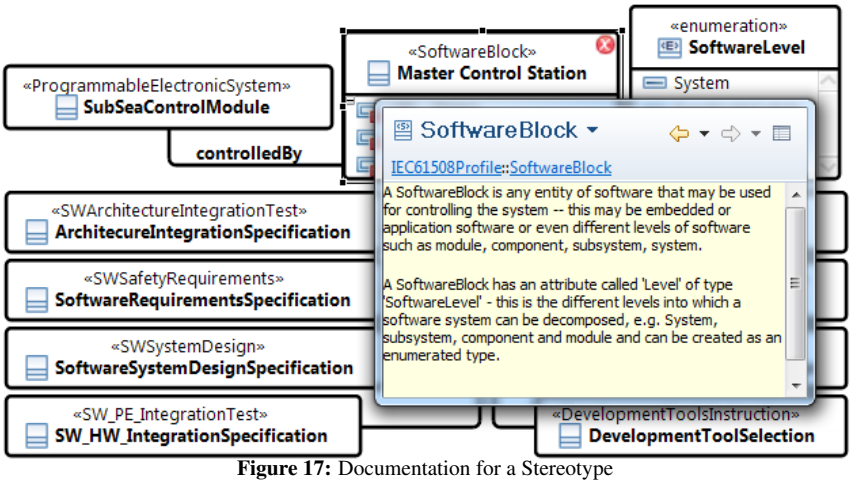


Figure 17: Documentation for a Stereotype

determine the feasibility of our approach to support certification and the survey helps assess whether domain experts see benefit in adopting our approach in a real industrial context.

5.1 Case Study

Our case study is aimed at investigating the feasibility of our approach and the level of effort involved in its application. Below, we provide a detailed description of the context, execution, and outcomes of the case study.

5.1.1 Nature of the Case Study

The subject of the case study is a new approach for improving upon the current practice of safety certification. Our case study can therefore be seen as an *improvement case study* as described by Runeson et. al. [34]. The case study has been conducted in an industrial setting and is a means for showing that the proposed approach is viable for use in industry.

5.1.2 Research Questions

The case study is targeted at answering the following research questions:

- **RQ1. Is the approach feasible?** More specifically, this question is concerned with (1) whether it is possible to represent the evidence requirements of a safety standard in terms of a conceptual model, and (2) provided that the answer to the first part is positive, whether it is possible to encapsulate the guidance for the creation of certification evidence into a UML profile based upon a standard's conceptual model. For answering RQ1, we do not concern ourselves with the creation of the domain models and instance models envisaged in our approach. These activities do have implication on the effort involved in carrying out the case study (see RQ2) but are technically well-understood in practice and do not require a feasibility study.

- **RQ2. Is the effort involved in the application of the approach acceptable?** The answer to this question is based on the level of effort spent throughout the case study. Effort is an important factor for the successful adoption of a new approach. If practitioners do not find the level of effort reasonable, they are unlikely to adopt the approach.

5.1.3 Case Selection

Our approach was motivated by the issues that our industrial partners faced during safety certification, both on the side of the certifiers and the suppliers of safety-critical systems. To apply the approach in an industrial setting, we needed two prerequisites to be in place (1) access to a safety-critical system that has undergone safety certification recently, is currently being certified, or is about to be certified in the near future. Old certification projects were deemed unsuitable for a case study due to the difficulty of acquiring sufficient details about them. (2) access to domain experts and securing adequate participation from them for the case study. We note that safety certification is a necessary but relatively infrequent event: entirely new safety-critical systems that need be certified are rare and the existing systems evolve rather slowly and require re-certification once every few years. Due to the scarcity of cases, we had to be opportunistic with case selection, as long as the two prerequisites above were satisfied.

The system suppliers that we had access to were involved with the certification of sub-sea production systems (discussed earlier in Section 3). The timing of our case study coincided with the construction of a new oil field, whose sub-sea production system needed to be certified in the near future. Within this system, we still had to choose a specific part to work on as these systems are very large and performing a case study on a complete system would have required resources beyond what was available at the time. Since we are primarily interested in software safety certification, we chose to concentrate our study on the software aspects of the sub-sea production control system for the new oil field. More specifically, the goal of the case study was to determine whether the software development plan being used by the supplier complied with the certification requirements. The software development plan outlines the activities that are carried out during software development and the resulting artifacts that are used as evidence during certification to show compliance. Our aim was to help the supplier determine which artifacts to create during the development of the software for the new control system.

The certification standard that the supplier needed to comply with was the IEC61508 standard [15] which sets forth the certification requirements for control systems that incorporate both mechanical and electronic components controlled by software. The aim of the standard is to ensure that safety-critical systems operate correctly in response to their inputs and that the system is brought to a safe state should a hazardous situation occur - known as *functional safety*. This standard is a large and comprehensive generic standard that is utilized in many domains, making it a good indicator for the feasibility of our approach. Moreover, it has been specialized for a number of domains such as the process industry [14], railways [13], automobiles [17] and others. In this sense, being able to apply our approach successfully for this standard is a good indicator of the generalizability of our work.

5.1.4 Data Collection Procedure

The procedure taken for conducting the case study closely followed the approach described in Section 3. In the first step, a conceptual model was built for the IEC61508 standard by analyzing the text of the standard. This work was done by the first two authors. The model was subsequently reviewed by an expert specializing in certification based on IEC61508 and subsequently was revised. The revised model was then presented to a group of twenty-eight certification experts in an industry workshop. During this workshop, the modelling notation was explained and the model itself fully presented. A question and answer session was held. This session resulted in no further changes to the model.

The second step was the creation of the IEC61508 profile. This was carried out by the first author. The basis for the profile is the conceptual model of IEC61508 built in the first step. In addition, OCL constraints were added to the profile, to provide guidance in elaborating the domain model.

The third step was the elaboration of the domain model. Since we did not have a domain model for sub-sea control systems a priori, we had to develop one. To this end, we first created a generic domain model for these systems using a general description of the systems found in [3, 16]. We then reviewed and refined this model over several meetings with a domain expert in the sub-sea domain at the supplier company where we were conducting our case study. The resulting domain model is one specialized to the needs of the supplier and includes concepts specific to the sub-sea control systems developed by the supplier. Once the domain model had been created, the elaboration process was carried out. As mentioned earlier in Section 5.1.3, a complete elaboration was not possible with the resources available to the supplier, hence the elaboration was carried out for the parts that were most relevant to the system supplier: the software development process - the activities and artifacts mentioned in the software development plan used by the supplier were modelled and the elaboration process carried out to check if these were sufficient or whether other activities and artifacts needed to be added. The fourth and final step was the creation of an instance for the particular system that was being certified. Steps 3 and 4 were carried out by the first author and the results reviewed by the experts at the partner company.

5.1.5 Results

In this section, we present the results of our case study. Parts of the case study were used to explain the approach in Section 3. In this section we concentrate on providing an overview of the outcomes of the case study without repeating any technical details that have already been discussed in Section 3 along with examples from the case study.

Step 1 (Conceptual Model of IEC61508). The IEC61508 standard consists of seven parts of which parts one, two and three contain the requirements for the functional safety of the system. Each of these parts describes an overall safety life-cycle to achieve the required level of safety, part one for the overall system, part two for the hardware components, and part three for the software components. Parts four, five and six contain supplementary material such as definition and explanations of abbreviations, examples of methods for the determination of safety integrity levels and an overview of the measures and techniques that can be used to show the different levels of safety that have been achieved. We went through the whole

Table 2: Summary of Conceptual Model for IEC61508

Number of pages of IEC6108 closely examined	211
Number of textual requirements examined from IEC61508	318
Number of concepts extracted from the textual requirements	95
Number of UML packages created for grouping concepts	10
Number of relationships extracted from the textual requirements	51
Number of enumerations extracted from the textual requirements	8

standard first and then again with an emphasis on the requirements for functional safety (i.e., parts one, two and three of the standard) to create the conceptual model. The full model along with the associated glossary is given in Appendix 1.

In Table 2, we provide some statistics about the IEC61508 conceptual modeling activity and the contents of the resulting model. The standard is expressed as numbered requirements. A requirement in this case is a numbered item that expresses some criteria that must be met for a system to comply with the standard. We examined 318 such textual requirements and extracted 95 concepts of importance that are linked together by 51 relationships. We grouped these concepts into 10 packages encompassing related concepts. This conceptual model of the IEC61508 standard is the basis of the profile described in the next section.

Step 2 (UML Profile of IEC61508). In this step, all the concepts and relationships from the conceptual model of step 1 were transformed into stereotypes in a profile. We then augmented the profile with OCL constraints to provide guidance for elaborating the domain models. In Table 3, we provide a summary of the contents of the resulting UML profile. The number of stereotypes that extend the metaclass `Class` is the same as the total number of concepts in the conceptual model and the number of stereotypes that extend metaclass `Association` is the same as the number of associations in the conceptual model. We then have stereotypes extending the metaclasses `Model` and `Package` for helping to organize the model elements and finally their are OCL constraints for elaboration; recall from Section 3.2 that we have five different types of constraints, we show the number of each type in Table 3. This profile is then used for the elaboration process described in the next section.

Step 3 (Elaborating the Domain Model of Sub-sea Production Systems). The domain model of the system was made in close consultation with experts in a large maritime and energy company and based on a reading of the relevant literature where the architecture and the components of sub-sea systems (including the control software operating on them) are described [3, 16, 22, 36]. A fragment of the high-level breakdown of sub-sea production systems was shown in Figure 6. The complete high-level model is shown in Figure 49 in Appendix 1. After creation of the high-level domain model, we then concentrated on modeling the sub-sea production control system. We modelled only a small part of this system resulting in a model with forty-six elements and sixty-two relationships in it, of which twelve of the elements were subsystems requiring further breakdown but were not in the scope of the case-study. We do not show this model here as it contains proprietary information from our supplier, however, we did present a small sanitized fragment of it in Figure 7 (Section 3.3).

Table 3: Summary of Profile Stereotypes for IEC61508

Number of stereotype extending metaclass `Model`	1
Number of stereotype extending metaclass `Package`	4
Number of stereotype extending metaclass `Class`	95
Number of stereotype extending metaclass `Association`	51
Total number of OCL constraints in profile	218
Number of Type 1 OCL constraints in profile	95
Number of Type 2 OCL constraints in profile	53
Number of Type 3 OCL constraints in profile	42
Number of Type 4 OCL constraints in profile	20
Number of Type 5 OCL constraints in profile	8

The elaboration of the domain model began with applying the stereotype `IEC61508Model` to the domain model. We then continued with applying the system stereotypes and subsequently concentrated on the software control system to determine whether the activities and artifacts mentioned in the software development plan used by the supplier were in line with those prescribed in the standard. The process was as described in Section 3.3. We added all the artifacts that were needed to satisfy the constraints for the software.

When artifacts are added to a model, their constraints will require the addition of the activities that generate those artifacts. This led to the creation of the complete process model for the software development. Once all the constraints were satisfied for the software, we compared the activities and artifacts created during the elaborated model with those that the supplier had as part of their software development plan. The supplier was keen to check how well they satisfied the requirements of the IEC61508 standard based on their current software development plan. We show a summary of this exercise in Table 4. The standard defines 16 activities that are carried out for software development and the requisite artifacts that are the output of these activities. From the 16 activities, the supplier had 10 of them defined in their software development plan and only 10 of the required artifacts were in the plan from the requisite 27. The supplier needed to update their software development plan and add the missing activities, artifacts and required competence and disseminate this information to their engineers.

Step 4 (Instantiating the Domain Model). The instance model was created for the software development process of a specific system. The model represented all the activities that would be carried out and the artifacts generated. It also highlighted that the developer needed to define the competence required for some of their activities. The instance diagram created for the software development is quite large as it defines all the activities for software development and the artifacts generated as evidence from these activities. Some of the activities are carried out multiple times, e.g., the module design activity will be performed for every module in the software, similarly with the module testing activity. Thus there will be as many artifacts specifying the module design, the module test specification and the results of testing, as the number of modules in the system (we cannot show this information due to confidentiality

Table 4: Supplier Software Development Plan Versus IEC61508 Requirements

Number of activities required by IEC61508 for software development	16
Number of activities in the supplier software development plan	10
Number of artifacts required by IEC61508 for software development	27
Number of artifacts in the supplier software development plan	10
Number of activities requiring definition of competence in the supplier's software development plan	6

reasons). All this information is captured in a central place and linked to the actual artifacts that will be necessary for certification (as shown in Figure 16 in Section 3.3). The numbers in Table 5 show the different types of activities to be carried out and the analogous number of artifacts. The actual number for a system will vary depending on how the system is broken into subsystems and modules.

5.1.6 Discussion

Below, we discuss the results of the case study focusing on answering the research questions that we presented in Section 5.1.2.

- **RQ1. Is the approach feasible?** We could successfully extract concepts and relationships from the IEC61508 standard based on the process described in Section 3.1. After finishing the process, we amalgamated the different diagrams into one main diagram that showed all the concepts and their relationships for the overall standard. For software development, we kept separate diagrams per software activity as shown in Appendix 1.

  While identifying the concepts, we tried as much as possible to use the terminology used by IEC61508. Specifically, we were trying to use the terminology defined in part four of the standard, but we found several terms that were used in the standard but not defined. For example, the terms `Activity`, `Competence`, `Enhancement` and `Issue` all appear in the text of the standard but no definition is given for them, the assumption perhaps being that these are naturally understandable to readers. However, providing a definition for each concept is an important prerequisite for creating an explicit interpretation of the standard and minimizing the possibility of ambiguity. For the terms used but not defined by the standard, we had to develop our own definitions based on both the context of their use and the definitions found in the literature on safety and reliability.

  Providing definitions for the relationships between concepts turned out to be a more challenging task, as the standard does not explicitly discuss the links between the concepts. The names for the relationships were chosen based on our reading of the text; however, we had to develop the definitions for the relationships on our own in the same manner as for the concepts that were not defined by the standard.

  After revisions based on comments from the certification expert, we presented the model at a workshop where the participating certification experts agreed that we had captured the salient concepts and relationships within the standard. An important issue

Table 5: Summary of Supplier Software Development Plan at the Partner Company

Number of activities to be carried out for software development	16
Number of artifacts to be produced during software development	27

to note here is that our model should not be viewed as a comprehensive interpretation of IEC61508. In particular, the standard contains numerous requirements concerned with how to perform the various lifecycle activities and how to describe the contents of the artifact, e.g., "Coding standards shall be a) reviewed as fit for purpose by the assessor; and b) used for the development of all safety-related software", and "The documentation shall be easy to understand by those having to make use of it". We do not capture such requirements in our conceptual model as we are concerned with the type of evidence required for certification and not the quality attributes of the evidence. At the most we can add this text to the documentation for the stereotype concerned, thus the user will see it when using the stereotypes and be aware of these requirements.

Overall, we observed that having a graphical representation of the conceptual model was very useful for presenting it. The certification experts and practitioners from the supplier agreed that the choice of UML class diagram for representing the conceptual model was useful and easy enough to understand. This finding is confirmed by the survey that we present in Section 5.2.

With regards to the construction of the profile for IEC61508, it was straightforward to translate the concepts and their relationships from the conceptual model into stereotypes. The challenging part was developing the OCL constraints. The constraints were not only used to establish the properties of the stereotypes themselves but to add additional information that would guide the user in applying the profile to domain models. All these constraints were added manually to the model. From the five types of constraints mentioned in Section 3.2, it is possible to automate the process of adding the constraints of type 2, 3, 4 and 5 as these are derived directly from the conceptual model based on the type of metaclass being stereotyped. This automation can take advantage of the fact that models can be queried and then software can be written to extract the required information and create the profile along with the OCL constraints. However, we chose to not pursue this route as this would have involved solving technical challenges as opposed to solving a research problem. Instead, we were more interested in assessing whether guidance could be provided for elaboration using OCL constraints. Constraints of type 1 were added for the software development process and required thought into what types of constraints would provide adequate guidance and which stereotypes the constraints should be added. Here, automation would not be possible as for types 2–5 since the constraints are not extracted directly from the conceptual model.

With regards to the domain model elaboration, we found the profile to be effective in guiding the construction of a relationship between the standard and the application domain. For the purpose of showing feasibility, we focussed on the elaboration of the artifacts created during software development. Even with this limited focus, the

supplier viewed this as a useful exercise, as they could more easily see which activities and artifacts were not part of their current software development plan. Some activities were missing even though they were carried out but their results were never explicitly documented for certification - this was the case for module design, which was carried out by the software engineers as part of the implementation work and never explicitly documented. Other activities were missing because they were not carried out as part of software development per se but rather as management activities - such as planning for the safety assessment of software. Thus, elaboration helped the supplier identify the diverse pieces of information that were relevant to software but were organized under headings other than software. All this information could now be linked to the software development plan and presented as such to the certification body at certification time. Furthermore, the supplier realized that they needed to link the competence data that they keep about their employees to the the activities of the developments process so as to show that the work was being performed by people (referred to as agents in Table 4) with the right competence. This information was also available but never linked to the certification information.

We note that our feasibility argument for domain model elaboration, as given above, focusses on a small fragment of the domain model, and in this case the number of constraints that are violated is manageable. However, we acknowledge that for a complete model there could potentially be a large number of violations that would need to be dealt with. In this case, it would be useful to have some form of prioritization of which constraints should be tackled first and which later. We do not propose a solution for this issue in this article and leave it for future work.

In summary, we recall that our focus for showing feasibility was the construction of the conceptual model of a standard and its use in creating a UML profile to provide guidance for the creation of certification evidence. To this end, we have shown that both aspects are feasible in a realistic certification context. As we would like our approach to be adopted in industry, we cannot rely on feasibility alone: we need to also consider the effort required and that is the subject of RQ2, discussed below.

- **RQ2. Is the effort involved in the application of the approach acceptable?** The creation and refinement of the conceptual model required approximately 6 person months. This effort also encompasses the time we spent to familiarize ourselves with the IEC61508 standard, which is a substantial document in itself.

  The creation of the profile took approximately 3 person months. This includes the (one-time) effort taken to investigate how we could use OCL constraints for guiding domain model elaboration and implement the constraints in the RSA tool (discussed earlier in Section 3.2).

  The construction of the domain model took another 2 person months. Most of this time was spent understanding the domain and creating the domain model. We had access to domain experts but they were not available to create the models, only to check them and provide feedback. Thus, the first author spent time learning the domain of sub-sea control systems and reading the system documentation provided by the supplier in order to create the models which were then refined based on comments from the experts.

The process of elaboration and instantiation were both completed within one month. This is comparatively less effort than the other tasks and was partly due to the limited scope in steps 3 and 4. It should be noted though, that for the little effort that was spent on these steps, the supplier was very pleased with the findings that the elaboration gave rise to.

The largest level of effort in our study was spent on the creation of a conceptual model for IEC61508 (the underlying certification standard). We anticipate such conceptual modeling to require a sizable effort for other standards as well; however, this activity needs to be performed only once per standard, or standard revision. We expect the effort for the creation of a profile to be less on future applications, as we had to address several technical details in relation to using OCL constraints in our first application.

The next most effort-consuming task was the creation of the domain model of the system which was done by a researcher unfamiliar with the domain or its terminology. The effort may be reduced if the models are created by someone familiar with the domain. Note that this effort is only needed once for a domain.

The experts felt that the initial effort for creating the conceptual model of the standard and the profile was a little high, however, since it is a one-time task, it was acceptable. The justification for this was that, given the number of projects that the supplier company would have to certify using the IEC61508 standard in the future, this effort would pay for itself over time. The creation of the domain model was a useful exercise to the supplier as well. This effort too can be spread over a number of projects, as the same model is used for all sub-sea production systems. Thus including a high amount of re-use in the steps of the approach has been a useful choice.

5.2 Survey

Following the completion of our case study, we conducted a workshop where we presented our solution and collected feedback from practitioners through a survey. In this section, we discuss the design and results of this survey.

5.2.1 Data Requirements

While preparing for the survey, we had to consider what factors would be of interest to the practitioners in assessing whether to adopt a technology, as well as what we needed to ask our participants in order to obtain useful feedback. We found the factors in Rogers' theory of innovation diffusion [33] highly relevant to consider:

- *Trialability* is the degree to which the technology can be tried on a limited basis or adopted in increments.
- *Compatibility* is the degree to which the technology is perceived to be consistent with existing values, experiences and needs of the practitioners.
- *Relative advantage* is the degree to which the new technology is perceived to be better than what is currently used.

- *Observability* is the degree to which the results of using the technology would be visible to others. This is important as visibility kindles discussion amongst peer groups and helps the spread of the technology. In our case, we would want visibility to the other members of the team and the certification body as well. This would then help spread the use of the technology.

- *Complexity* is the degree to which a technology is perceived to be difficult to understand or use.

All the above factors are based upon the *perceptions* of the practitioners. There may be ways to objectively measure the relative advantage of a technology or its compatibility and so forth but it is the perceived advantage in terms of a particular factor that matters most for the adoption of the technology.

The case study described in Section 5.1 enabled us to try our approach in a small but realistic setting, providing a suitable context to examine *trialability*. With regards to *compatibility*, we knew beforehand that our solution presented something different from what the practitioners were accustomed to. While the solution was developed in response to the practioners' direct needs, we have not yet attempted to integrate it into the current workflow of certification activities at the partner company. Therefore, we cannot at the moment address compatibility in a direct way. The same is true for *relative advantage* - we did not want to compare the current mode of working with our solution as we did not have any means to show comparison. Instead of asking direct questions about compatibility and relative advantage, we decided to ask a question about whether the experts would see value in adopting our approach.

As for *observability*, we note that the approach is collaborative by design and would entail an evolution of work practices, thus contributing to visibly within the entire team. In fact, the execution and the outcomes of our case study although limited in scope, already attracted considerable attention at the partner supplier. Further and in relation to observability, we wanted to investigate if the partner supplier would be willing to use the developed models for interaction with the certification body as well, in turn increasing the observability of the approach. We have a question to this end in the survey. Finally, our survey has questions to directly assess the perceptions of the participants about the *complexity* of the approach.

5.2.2 Data Generation Method

To carry out the survey, we needed to first provide a reasonably thorough overview of our solution to the experts. This was done in an interactive workshop lasting for two hours. We wanted to present our work to as many relevant practitioners at the supplier company as possible. Our sampling frame was practitioners involved with developing software for safety critical systems that are to be certified by a third-party certification body. Within this sampling frame, we used the snowball sampling technique [23] combined with purposive sampling [23] to select the participants for the survey. In snowball sampling, one person is picked from the target sample frame, and once data has been gathered from this person, she is asked to recommend others to contact for further data collection. This process can be repeated to add further data samples. In purposive sampling, the sample group is hand-picked to include those who would likely produce the most valuable data for the purpose of the research. In our case, we had a champion who had been working with us throughout the case study and was familiar

Q1. Is certification an important aspect of your job?

- ☐ Yes
- ☐ No

Q2. How much experience do you have with certification-related activities?

- ☐ Less than 6 Months
- ☐ More than 6 months but less than 12 months.
- ☐ More than 1 year but less than 2 years
- ☐ More than 2 years

Figure 18: Questions 1 and 2 of the survey

with both the purpose and technical details of the work. We asked her to contact all staff members she knew of, who satisfied our sampling frame criterion. The champion sent out an email invitation to twenty-three staff members including developers, project managers and product managers. The email informed the invitees that a model-driven approach for IEC61508 certification would be presented. After this, neither we nor our champion had any control over the sample - we surveyed those who came to the workshop.

During the workshop, we first explained all the steps of our approach, providing examples drawn from the partner company's application domain. We then presented the results of our case study, followed by a demonstration of tool support. There was time for questions so that we could clarify any concerns. After the question and answer session, we circulated a questionnaire for the participants to answer. The questionnaire was anonymous, although participants were given the option to provide their name and contact information in case any follow-up was required - the choice was up to them to participate. Due to the strict time constraints that we had to observe, we had to be very selective with the questions that we put on the questionnaire – we needed to cover our data requirements while ensuring that the questionnaire was clear and succinct.

5.2.3 The Questionnaire

The questionnaire was divided into four parts. The first section, Q1–Q2 shown in Figure 18, was about the background of the subjects related to certification in general. We explained in the questionnaire that "certification-related experience" covers the following: (1) Attending tutorials and workshops on certification processes and standards; (2) Self-reading of certification standards; (3) Attending certification meetings; (4) Constructing and reviewing certification reports and deliverables.

The second section, Q3–Q6 shown in Figure 19, was about the participants' experience with the IEC61508 standard for certification. Although we had based our work on prior knowledge about the difficulty in using text-based standards, we wanted to ensure that this was also the case in this group. For this set of questions, if the answer to Q3 was "No", the participant was to skip Q4–Q6.

The third section, Q7–Q8 shown in Figure 20, was about the modelling of safety standards. In particular, we wanted to find out what the participants thought about modelling of a textual standard.

Q3. Is your own work (current or past) related to demonstrating compliance to IEC61508?

- ☐ Yes
- ☐ No

Q4. Have you read the IEC61508 standard?

- ☐ Entirely
- ☐ To a Great Extent
- ☐ Somewhat
- ☐ Very Little
- ☐ Not At All

Q5. Based on your experience, how easy to understand is the text of the IEC61508 standard?

- ☐ Very Easy
- ☐ Easy
- ☐ Average
- ☐ Difficult
- ☐ Very Difficult

Q6. Based on your experience, is the IEC61508 standard easy to use for certification?

- ☐ Always
- ☐ Usually
- ☐ About half the time
- ☐ Seldom
- ☐ Never

Figure 19: Questions 3 to 6 of the survey

Q7. Was the presented conceptual model easy to understand?

- ☐ Very Easy
- ☐ Easy
- ☐ Average
- ☐ Difficult
- ☐ Very Difficult

Q8. If given a conceptual model of a standard like the one we presented, would you use that model to help in understanding the standard?

- ☐ Definitely
- ☐ Very Probably
- ☐ Probably
- ☐ Possibly
- ☐ Probably Not
- ☐ Very Probably Not

Figure 20: Questions 7 and 8 of the survey

Q9. How easy to follow were the steps in our approach?

- ☐ Very Easy
- ☐ Easy
- ☐ Average
- ☐ Difficult
- ☐ Very Difficult

Q10. Would you see value in adopting the presented approach at Company A for certification?

- ☐ Definitely
- ☐ Very Probably
- ☐ Probably
- ☐ Possibly
- ☐ Probably Not
- ☐ Very Probably Not

Q11. Do you find the models simple enough to use for communication with a certification body?

- ☐ Definitely
- ☐ Very Probably
- ☐ Probably
- ☐ Possibly
- ☐ Probably Not
- ☐ Very Probably Not

Q12. Does the presented tool provide useful assistance for certification at Company A?

- ☐ Definitely
- ☐ Very Probably
- ☐ Probably
- ☐ Possibly
- ☐ Probably Not
- ☐ Very Probably Not

Figure 21: Questions 9 to 12 of the survey

The final section, Q9–Q12 shown in Figure 21, was aimed at obtaining feedback about the overall approach and using model-driven engineering. For confidentiality reasons, we do not reveal the name of the collaborating company and refer to it as "Company A" in Q10 and Q12.

5.2.4 Survey Results

Out of the twenty-three invitees, twelve attended the workshop, yielding a response rate of approximately 52%. All groups (i.e., developers, project managers and product managers) were represented by those in attendance.

Based on the responses obtained, certification was an important aspect of the job for all but

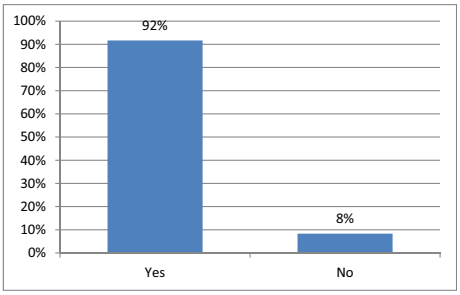


Figure 22: (Q1) Is certification an important aspect of your job?

one participant (Figure 22). This participant was a consultant, working as a project manager who would be engaged in certification activities in the future and hence had chosen to attend the workshop. Overall, 58% of the participants had over two years of certification experience and a further 17% had at least one year of experience with certification (Figure 23).

Regarding the experience of the participants with the IEC61508 standard, all except two had worked with this standard (Figure 24). From the two that had not worked with the standard, one was the consultant mentioned earlier while the other had experience with certification using a standard other than IEC61508. Moreover, certification was an important aspect of his work and he had attended the workshop because he was interested in the application of model-driven engineering to other standards.

From the set of participants who had used the IEC61508 standard, 40% had read the entire standard and a further 20% had read almost the entire standard. The rest had read some parts of it (Figure 25). Half of these participants felt the standard was difficult to understand and the rest thought it was of average difficulty (Figure 26). With regards to ease of use, 40% of the participants who had read the standard indicated that IEC61508 is seldom easy to use for certification, while 30% found it easy half the time; only 30% agreed that the standard is usually easy to use (Figure 27). When the findings shown in Figures 24 and 27 are taken together, we can conclude that 83% of the participants need to use the standard as part of their work and yet only 30% of them found the standard easy to use. This lends support to further research targeted at making large standards like IEC61508 easier to use for practitioners. Note that for Q5 and Q6 (Figures 26 and 27), we presented results from the subset of participants that had used the IEC61508 standard for demonstrating compliance, as determined by Q4 (Figure 25). The remainder of the results are from the entire group of participants.

When presented with the conceptual model of the IEC61508 standard, 8% of the participants found it very easy to understand and a further 67% found it easy (Figure 28). A further 17% thought the conceptual model was averagely easy to understand and 8% found it difficult. If we compare this to the ease of understanding of the textual standard, we find that 92% of the participants found the conceptual model to be very easy, easy, or averagely easy to understand compared to 50% finding the textual standard averagely easy while the other 50% finding it difficult to understand. Figure 29 shows that the participants unanimously agreed that they would probably use the model to help them better understand the textual standard. This shows that creating a conceptual model of a standard can be useful even when not employed as part of a wider certification strategy.

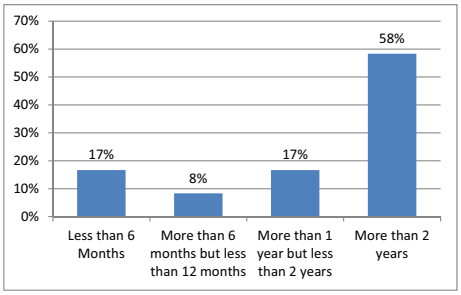


Figure 23: (Q2) How much experience do you have with certification-related activities?

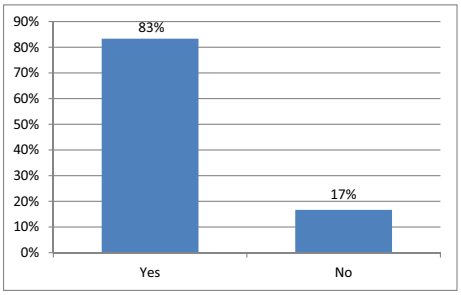


Figure 24: (Q3) Is your work (current or past) related to demonstrating compliance to IEC61508?

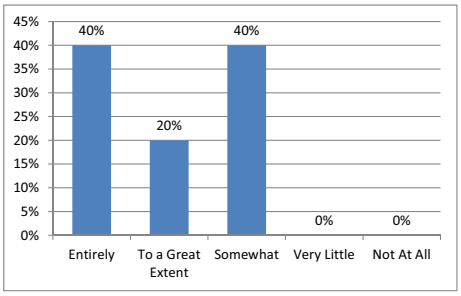


Figure 25: (Q4) Have you read the IEC61508 standard?

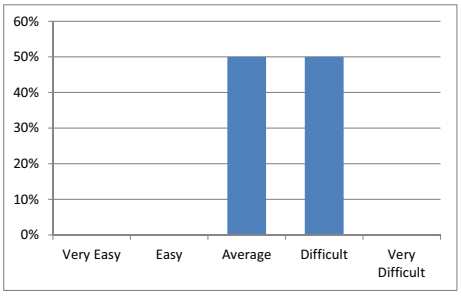


Figure 26: (Q5) Based on your experience, how easy to understand is the text of the IEC61508 standard?

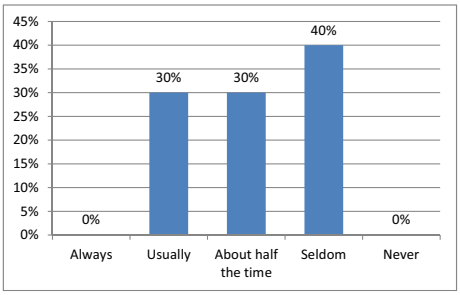


Figure 27: (Q6) Based on your experience, is the IEC61508 standard easy to use for certification?

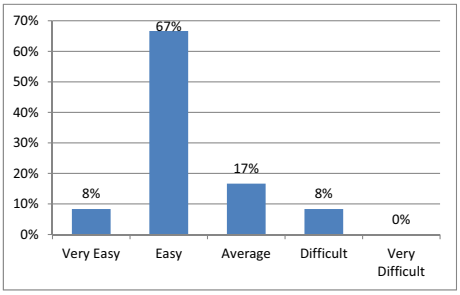


Figure 28: (Q7) Was the presented conceptual model easy to understand?

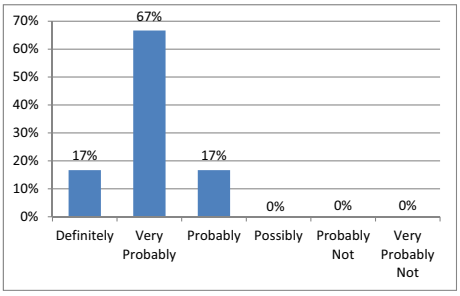


Figure 29: (Q8) If given a conceptual model of a standard like the one we presented, would you use that model to help in understanding the standard?

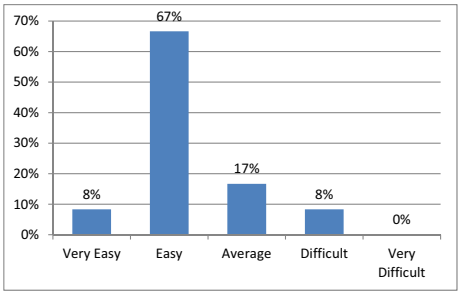


Figure 30: (Q9) How easy to follow were the steps in our approach?

The final four questions concern our approach as a whole. As stated earlier when explaining the questionnaire, we cannot disclose the name of our industry partner and refer to it as "Company A". We need to note though that the industry partner is a large supplier of safety-critical systems, with many of its systems subject to certification standards, IEC61508 being one of the key ones. Thus, they were a good representative company for evaluating our approach.

We found that 8% of the participants perceived the approach as being very easy to follow, while 67% thought it was easy, and a further 17% thought is was averagely easy to follow; no one found the approach difficult to follow (Figure 30). In terms of adoption, 42% of the participants thought that there was definitely value in adopting the approach and a further 50% thought the approach was very probably worth adopting. Thus, 92% of the participants were in favour of adopting the approach. The remaining 8% were not negative either and thought the approach was probably worth adopting as well (Figure 31). All participants agreed that the models created during the application of a model-driven certification approach were simple enough to use in communication with the certification body (Figure 32). The current implementation of the approach using IBM Rational Software Architect [12] was also thought to be useful: 33% of the participants believed that the tool would definitely be useful for certification and a further 42% thought it would very probably be useful. The remaining 25% of the participants were spread between probably useful (17%) and possibly useful (Figure 33). No negative opinions were expressed about the tool.

In summary, the answers suggest that the approach was overall viewed to be easy to understand and the participants thought that it would be advantageous to use it within their context.

6 Related Work

There are two main areas of work related to the approach presented in this article: managing certification evidence electronically and the use of UML for the development of safety-critical systems.

The need for constructing certification evidence electronically has been identified by [7] and [19] in order to manage the complexity and large amounts of information that needs to be collated. Lewis [19] calls for an underlying information model to manage the complex links

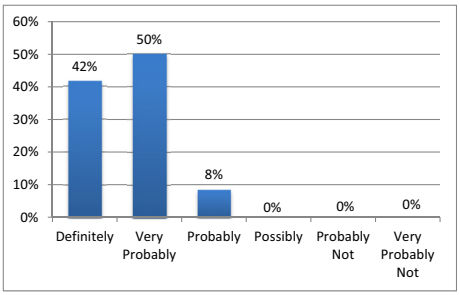


Figure 31: (Q10) Would you see value in adopting the presented approach at Company A for certification?

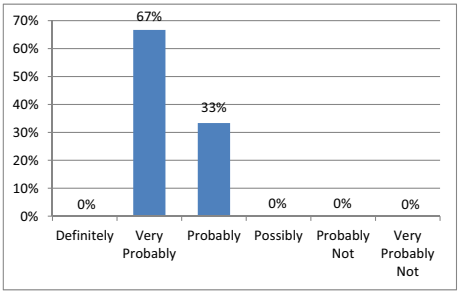


Figure 32: (Q11) Do you find the models simple enough to use for communication with a certification body?

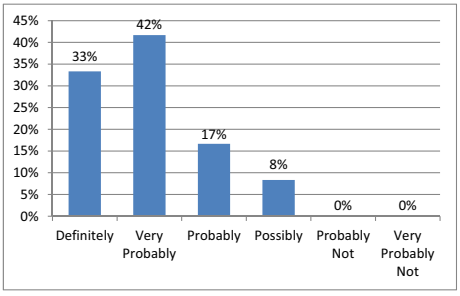


Figure 33: (Q12) Does the presented tool provide useful assistance for certification at Company A?

that exist between the various pieces of safety evidence. We propose an approach that takes into account both the above points. Our conceptual model provides the underlying information model for a standard and our profile provides a practical mechanism for using this information in order to create the relevant artifacts for certification. Cockram and Lockwood [7] present a proprietary tool that uses hypertext for linking all the diverse pieces of information for certification. This enables them to link information but the links are manually created without an underlying information model to guide the creation of the evidence as we do. We use model-driven engineering technologies, and in particular UML profiles, to aid with the creation of this evidence for a particular standard. The profile can be exported and used in any UML modelling tool and the constraints provide guidance when elaborating the domain model. The set of inter-related information items can be seen via the elaborated models and the instance models use uniform resource locators to link to artifacts directly.

The use of model-based technologies, especially UML, is gaining pace in the development of safety-critical software. The Object Management Group (OMG) has standardized the UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE) [26] and the UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms (QFTP) [25]. Both these profiles are used for modelling the real-time and performance properties of safety-critical systems. Similary, Berkenkotter [4] and Hannemann [5] have created a profile for the railway domain that aids the design and verification of interlocking functionality. However, neither of these are meant to characterize the evidence requirements of a standard according to which safety-critical systems are certified.

A profile that deals with certain aspects of certification is proposed by Zoughbi et. al. [37]. Their profile enables the direct addition of certification information to software models for compliance with the RTCA DO-178B standard [31], used in commercial and military aerospace software. However, this profile is targeted at maintaining traceability between requirements, design and code, which is only a part of the recommendations of DO-178B. The profile that we propose deals with a complete standard and takes into account not only evidence regarding requirements and design but also the wide range of concepts related to the management of the development process in safety-critical systems.

Huhn and Hungar [11] discuss the proliferation of UML in the model-based development of safety-critical software. They propose a development process where models form an integral part of the development of a safety-critical system. However, they do concede that the use of models for the certification has not been adequately addressed. Our profile is a starting point for addressing this gap.

Recently, the OMG has put forward a proposal, called the Software Assurance Evidence Metamodel (SAEM) [27], for managing safety assurance evidence. SAEM is a standard-independent metamodel and directed towards linking the certification evidence to safety claims and the evaluation of these claims subject to the evidence. The approach that we propose uses a UML profile for characterizing the evidence of a specific standard. To perform the same task, the SAEM model will still require a definition of the specific evidence needed by a particular standard (perhaps based on a conceptual model as we have proposed). On the other hand, a profile of the SAEM could be incorporated into our approach and cover both the evidence requirements for compliance as well as the evaluation of the evidence to ensure that it is sufficient to substantiate the safety claims. Together, these could be means to further the field of model-based certification.

Regarding compliance to a specific standard, Chung et. al.[6] study the problem of compliance of a user-defined workflow with the activities prescribed in IEC61508. They check (process) compliance by comparing user-defined activities in an organization against models of the activities in the standard. This work is similar to what we propose in that the aim is to model compliance information; however, we go beyond the process aspects of a standard and provide an evidence information model for the entire standard which in turn is the basis of our profile that is used to manage certification evidence.

Finally, we have used UML profiles of safety related standards in prior work [29], where we ensure that a generic standard can be specialized for a particular domain in a systematic manner. In contrast to this current paper, profiles were used in [29] as a way to keep track of the relationships between two standards – a generic and a sector-specific one.

7 Conclusion and Future Work

In this article, we described an approach based on model-driven engineering principles and technology to specify and analyze the safety evidence required for compliance to safety standards. We start by establishing a sound relationship between a domain model of a safety-critical application and the evidence model of a certification standard. We do so by capturing the relevant standard as a conceptual model in the UML notation and using the resulting model as a basis for creating a UML profile. The profile is augmented with constraints expressed in the OCL language to aid system suppliers in systematically relating the concepts in the standard to the concepts in the application domain. The profile is then applied to a domain model of a safety-critical application, aiding system suppliers in clearly demonstrating how the development artifacts of their system fulfil the compliance requirements of a standard. The constraints enforced by the profile can be automatically checked by existing OCL constraint engines.

Developing conceptual models along with glossaries of definitions for standards help avoid the ambiguity issues that can exist in text-based standards. The elaboration phase in our proposed approach provides step-by-step guidance on how to align the concepts in the system to the relevant standard and create the relevant evidence items. The domain models allow different levels of abstraction to be expressed. Thus, we are able to provide an overall picture of the system and at the same time the flexibility to drill down to a specific part of the system as necessary, while being aware of the connections between the different pieces. The different levels of abstraction and the breakdown of the system via the domain models also proves useful in collaborative work. Each engineer can work on the part of the system assigned to them, and create and elaborate the models as necessary. The changes are visible to the other engineers but need not interfere with their work unless there is some overlap.

We have applied our approach in a pilot study in the context of sub-sea production control systems. The case study shows that our approach is feasible in a realistic environment and that it can provide useful guidance in relating the concepts in the domain of sub-sea production control systems to those of the IEC61508 standard. The case study further shows that the effort involved in applying our approach was acceptable. Since IEC61508 is a generic standard that applies to multiple domains, a successful application of our approach to it is a good indication of the usefulness and wider relevance of our work.

We have further conducted a survey of industry practitioners. The results show that half of the domain experts we surveyed found IEC61508 difficult to understand and only 30% found it usually easy to use for certification. When presented with a conceptual model of the standard, 92% of the participants found the conceptual model to be either very easy, easy, or averagely easy to understand while only 50% found the textual standard averagely easy to understand. When surveyed about our approach, no one found it difficult to follow the steps of the approach and 92% of participants were in favour of adopting the approach in their work.

In future work, we would like to add a report generator to present the models in the form of reports for upper management. This would allow the provision of relevant information for a number of different actors in an organization. The approach itself can be extended for the certification of a system to multiple inter-related standards. We would also like to study whether conceptual models can be used as means for presenting standards and whether the use of models would make it easier to identify and take action on changes in the standards when they are updated. Finally, we would like to extend our work to help the certification body with the evaluation of evidence collated according to our approach.

Appendices

1 The IEC61508 Conceptual Model

The conceptual model for the IEC61508 is shown in Figure 34 as a UML class diagram. The conceptual model has a total of ten packages, containing abstractions for modelling the main concepts of IEC61508. We briefly explain each package. For more details, see [30]. The `System Concepts` package describes the breakdown of the system and reflects both hardware and software concepts; the `Hazard Concepts` package captures the abstraction for describing the hazards and risks for the system and leads to the specification of safety requirements; the `Requirements Concepts` package captures the requirements for creating, operating, maintaining and decommissioning control systems; the `Process Concepts` package is for describing the development process for creating the system; the `Artifact Concepts` package is for describing the different types of artifacts created as supporting evidence during the development of the system; the `Guidance` package is for describing the other standards and recommended practices that will be used to develop the system, the `Issue Concepts` package is for describing the defects or enhancements that may give rise to changes; the `Configuration Management Concepts` package is for describing the unique versions for all the components that make up the system, the `Justification Concepts` package to capture the assumptions and rationale behind the various decisions that are made during development; and the `Domain-Specific Concepts` package for capturing the enumerations for concept attributes in other packages (e.g., requirement type, artifact state).

Along with the conceptual model, a glossary was created for each concept and relationship, a part of this glossary, describing the most important concepts is shown in Table 6.

Table 6: Description of Main Concepts from the IEC61508 Metamodel

Concept	Description
`Activity`	A unit of behaviour in a process.
`Agent`	A person or organization that has the capability and responsibility for carrying out an activity.
`Artifact`	One of the many kinds of tangible by-products produced during the development of a system.
`Assumption`	A premise that is not under the control of the system of interest, and is accepted as true without a thorough examination. Assumptions can, among other things, be related to the environment of the system, the users, and external regulations.
`Block`	Entity of hardware or software, or both, capable of accomplishing a specified purpose.
`Change`	A modification made to the PES, Block or Artifact.
`Competence`	The ability to perform a specific task, action or function successfully.
`ControlledItem`	A PES, Block or Artifact for which meaningful increments of change are documented and recorded.
Continued on next page ...	

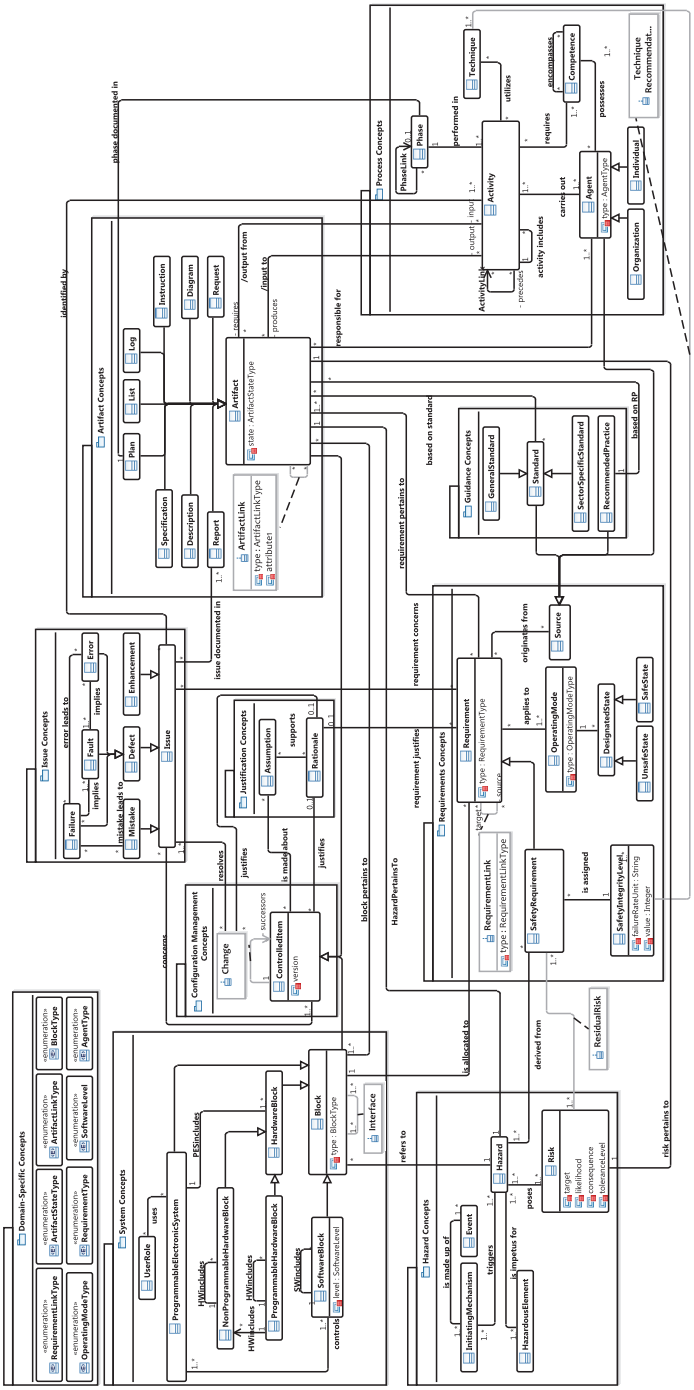


Figure 34: The IEC61508 Conceptual Model

Continued from previous page ...	
Concept	**Description**
`Defect`	An error, failure, or fault in a system that produces an incorrect or unexpected result, or causes it to behave in unintended ways.
`Description`	A planned or actual function, design, performance or activity (e.g., function description).
`DesignatedState`	The state of the EUC related to safety, the EUC is either in a safe state or an unsafe state.
`Diagram`	Specification of a function by means of a diagram (symbols and lines).
`Enhancement`	Provision of improved, advanced, or sophisticated features.
`Error`	Discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition.
`Event`	A single occurence in a series of occurences that cause a hazard to occur.
`Failure`	Termination of the ability of a functional unit to perform a required function.
`Fault`	Abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function.
`GeneralStandard`	A standard that provides generic recommendations on a specific subject to a number of related domains.
`HardwareBlock`	Any entity of hardware – this may be mechanical, electrical or electronic that is used in the composition of the system.
`HazardousElement`	The basic hazardous resource creating the impetus for the hazard, such as a hazardous energy source such as explosives being used in the system.
`Hazard`	Any real or potential condition that can cause injury, illness, or death to personnel damage to or loss of a system, equipment or property or damage to the environment.
`Individual`	Refers to a person.
`InitiatingMechanism`	The trigger or initiator event(s) causing the hazard to occur. The IM causes actualization or transformation of the hazard from a dormant state to an active mishap state.
`Instruction`	Specifies in detail the instructions as to when and how to perform certain jobs (for example operator instruction).
`Interface`	An abstraction that a block provides of itself to the outside. This separates the methods of external communication from internal operation.
Continued on next page ...	

Continued from previous page ...	
Concept	**Description**
Issue	A unit of work to accomplish an improvement in a system.
List	Information in a list form (e.g., code list, signal list).
Log	Information on events in a chronological log form.
Mistake	Human action or inaction that can produce an unintended result.
NonProgrammable-HardwareBlock	Electro-mechanical devices (electrical) solid-state non-programmable electronic devices (electronic).
OperatingMode	The different modes that a system can be operating in, e.g. normal, maintenance, test, emergency.
Organization	A social arrangement which pursues collective goals, which controls its own performance, and which has a boundary separating it from its environment.
Phase	A set of activities with determined inputs and output that are carried out at a specific time during the life of a system.
Plan	Explanation of when, how and by whom specific activities shall be performed (e.g., maintenance plan).
Programmable-ElectronicSystem	System for control, protection or monitoring based on one or more programmable electronic devices, including all elements of the system such as power supplies, sensors and other input devices, data highways and other communication paths, and actuators and other output devices.
Programmable-HardwareBlock	Any physical entity based on computer technology which may be comprised of hardware, software, and of input and/or output units.
Rationale	The fundamental reason or reasons serving to account for something.
RecommendedPractice	Sound practices and guidance for the achievement of a particular objective.
Report	The results of activities such as investigations, assessments, tests etc. (e.g., test report).
Request	A description of requested actions that have to be approved and further specified (e.g., maintenance request).
Requirement	A necessary attribute in a system; a statement that identifies a capability, characteristic, or quality factor of a system in order for it to have value and utility to a user.
ResidualRisk	Risk remaining after protective measures have been taken.
Risk	Combination of the probability of occurrence of harm and the severity of that harm.
SafeState	The state of the EUC when safety is achieved.
SafetyIntegrity-Level	The probability of a safety-related system satisfactorily performing the required safety functions under all the stated conditions within a stated period of time.
Continued on next page ...	

Continued from previous page ...	
Concept	**Description**
SafetyRequirement	A prescriptive statement that ensures that the system carries out its functions in an acceptably safe manner.
SectorSpecific-Standard	A standard that provides recommendations for a specific industrial sector (e.g., the energy sector).
SoftwareBlock	Any entity of software that may be used for controlling the system – this may be embedded or application software or even different levels of software such as module, component, subsystem, system.
SoftwareLevel	The different levels into which a software system can be decomposed, e.g. System, subsystem, component and module.
Source	An abstract concept that can represent a person, organization or standard that can be a source of requirements to a system.
Specification	Description of a required function, performance or activity (e.g., requirements specification).
Standard	An established norm or requirement, typically provided as a formal document that establishes uniform engineering or technical criteria, methods, processes and practices.
Technique-Recommendation	A particular technique recommended based on the safety integrity level of the requirements that have been allocated to the block in question.
Technique	A procedure used to accomplish a specific activity or task.
UnsafeState	The state of the EUC when safety is compromised.
UserRole	An aspect of the interaction between a PES and the human elements.

1.1 Software Lifecycle Activity Packages

The IEC 61508 standard prescribes certain safety lifecycle activities for software. In this section, we show how we have modelled the activities recommended for the software lifecycle along with the input and output artifacts of each activity.

Software Safety Requirements Definition The goal of this activity to specify the requirements for software safety in terms of the requirements for software safety functions and software safety integrity. See Figure 35.

Software Safety Validation Planning The goal of this activity is to develop a concrete plan for validating the software in terms of safety. See Figure 36.

Software Architecture Development The goal of this activity is to create a software architecture that fulfils the specified requirements for software safety with respect to the required safety integrity level. See Figure 37.

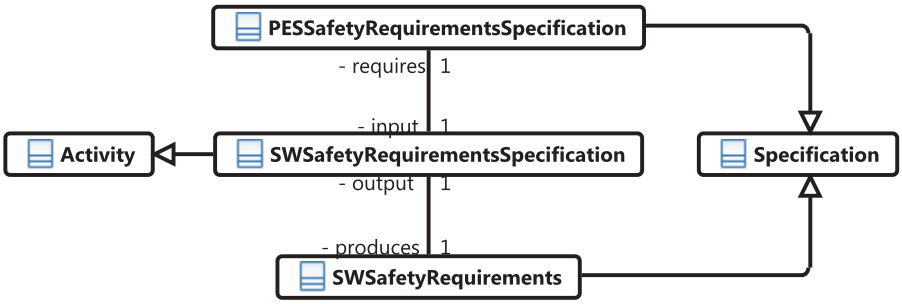


Figure 35: Software Safety Requirements Definition

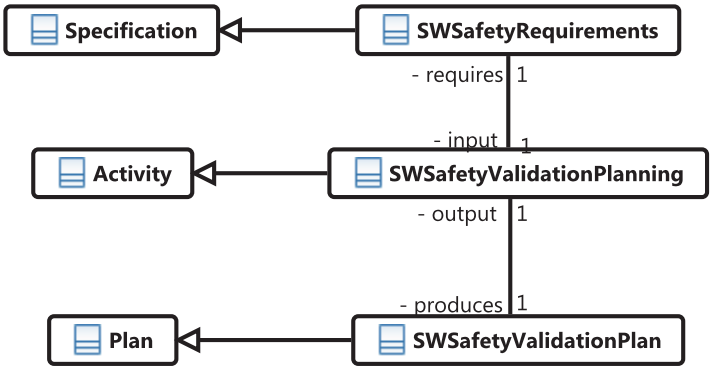


Figure 36: Software Safety Validation Planning

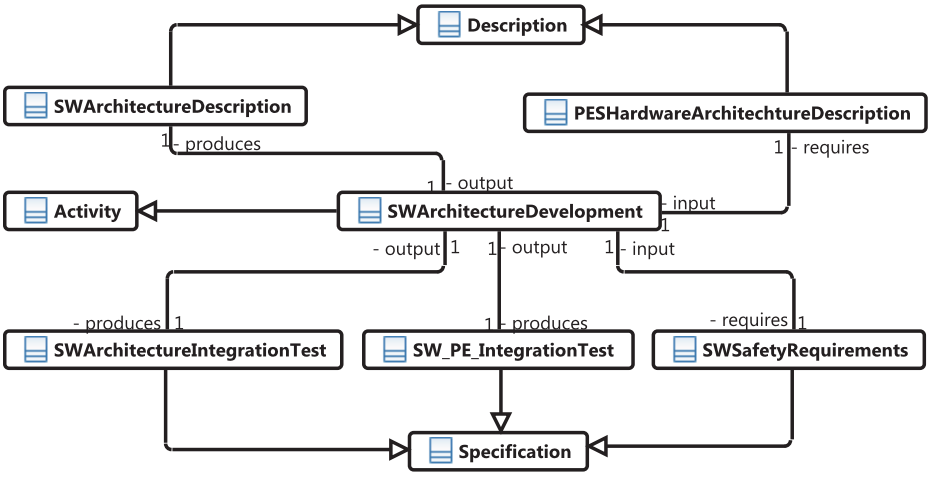


Figure 37: Software Architecture Development

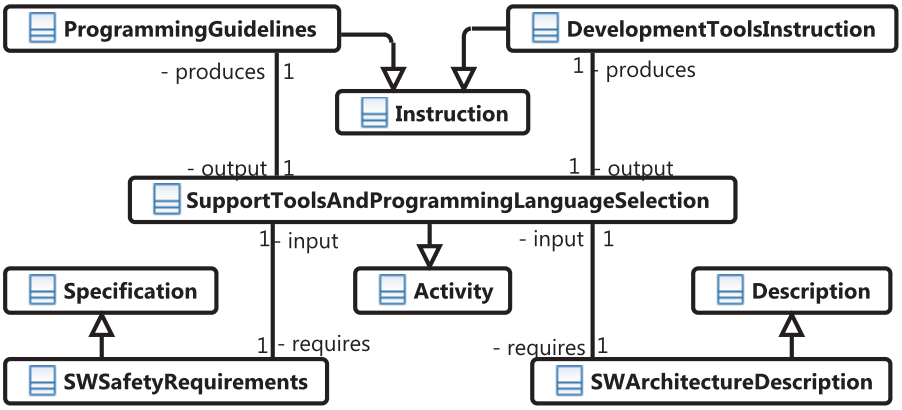


Figure 38: Support Tools and Coding Standard Development

Support Tools and Coding Standard Development The goal of this activity is to select a suitable set of tools, including languages and compilers, for the required safety integrity level over the whole safety lifecycle of the software. See Figure 38.

Software System Design Development The goal of this activity is to design software that fulfils the specified requirements for software safety with respect to the required safety integrity level, which is analysable and verifiable, and which is capable of being safely modified. See Figure 39.

Software Module Design Development The goal of this activity is to design the software modules that fulfill the specified requirements for software safety. See Figure 40.

Source Code Implementation The goal of this activity is to implement software that fulfills the specified requirements for software safety. See Figure 41.

Software Module Testing The goal of this activity is to verify that each software module performs its intended function and does not perform unintended functions. See Figure 42.

Software Integration Testing The goal of this activity is to verify that all software modules, components and subsystems interact correctly to perform their intended functions and do not perform unintended functions. See Figure 43.

Programmable Electronics Integration The goal of this activity is the integration of the software onto the target programmable electronic hardware and to combine the software and hardware in the safety-related programmable electronics system to ensure their compatibility and to meet the requirements of the intended safety integrity level. See Figure 44.

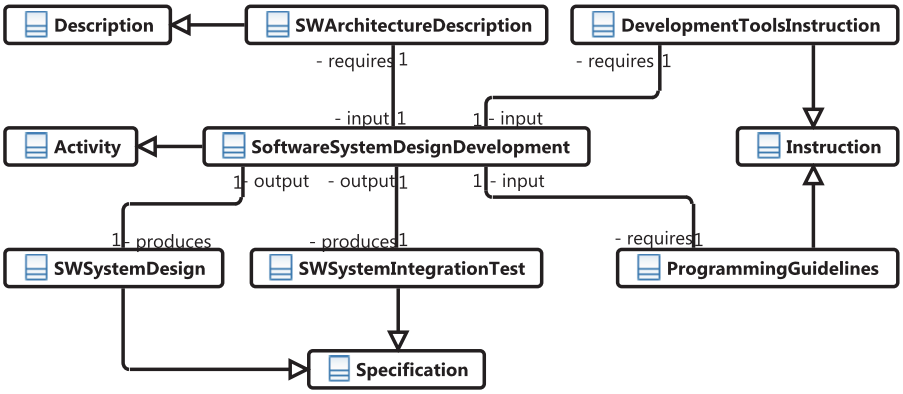


Figure 39: Software System Design Development

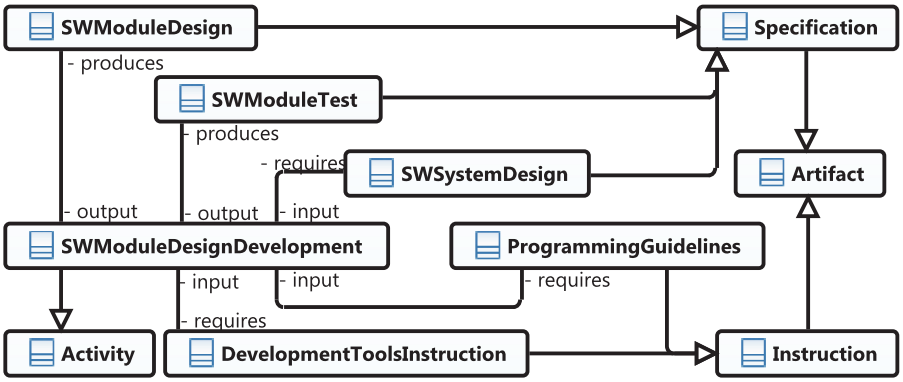


Figure 40: Software Module Design Development

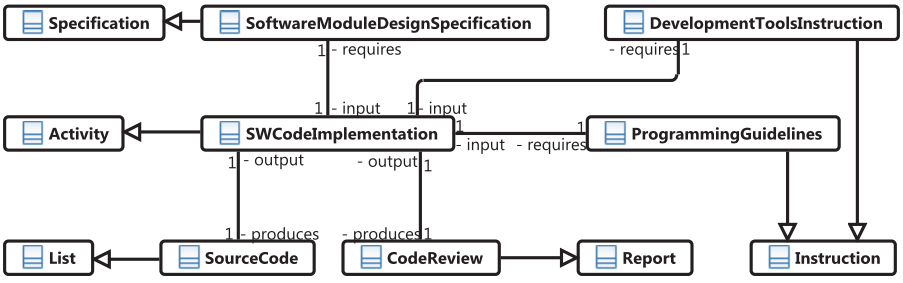


Figure 41: Source Code Implementation

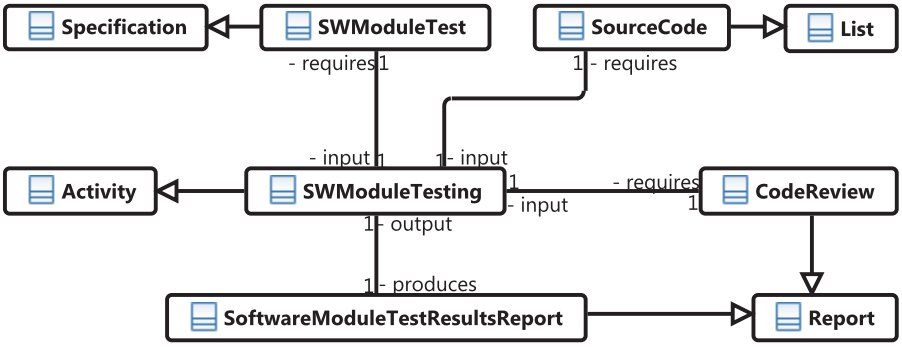


Figure 42: Software Module Testing

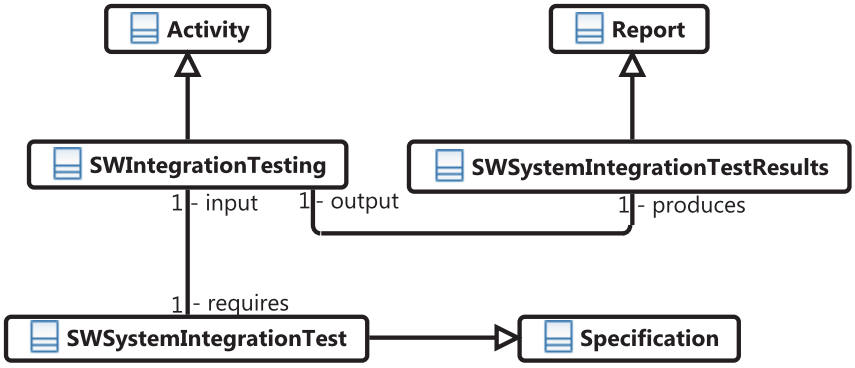


Figure 43: Software Integration Testing

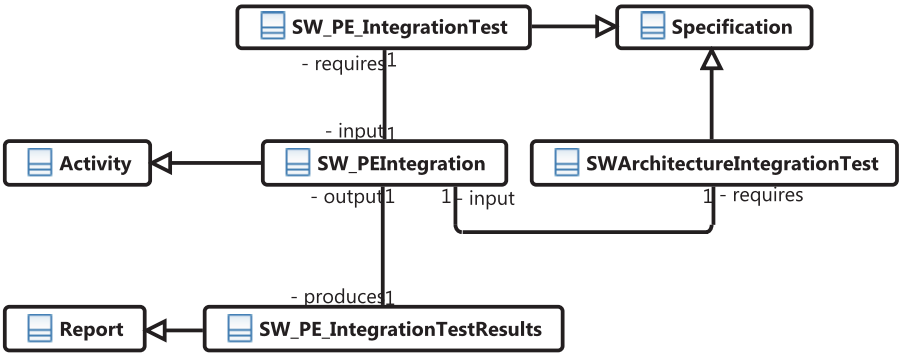


Figure 44: Programmable Electronics Integration

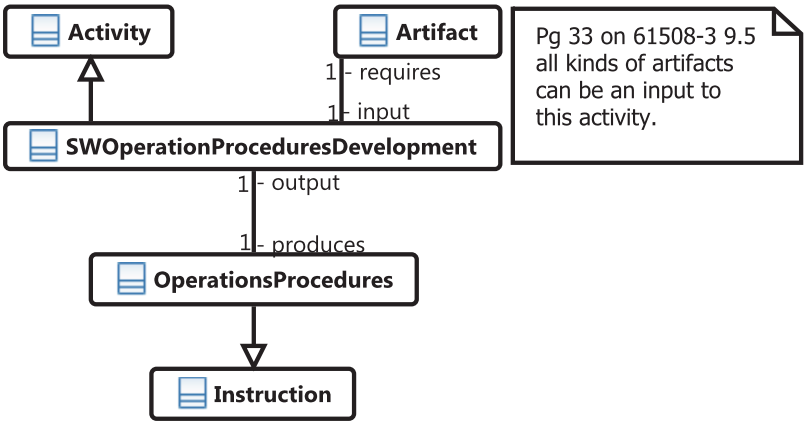


Figure 45: Software Operation Procedures Development

Software Operation Procedures Development The goal of this activity is to provide information and procedures necessary to ensure that the functional safety is maintained during the operation of the system. See Figure 45.

Software Modification Procedures Development The goal of this activity is to provide information and procedures necessary to ensure that the functional safety of the system is maintained during modification of the software. See Figure 46.

Software Safety Validation The goal of this activity is to ensure that the integrated system complies with the specified requirements for software safety at the intended safety integrity level. See Figure 47.

Cross-Cutting Activities: Software Modification, Verification, and Functional Safety Assessment These three activities (shown in Figure 48) link to all other activities being performed and can thus potentially affect all of them:

- *Software Modification:* The purpose of this activity is to ensure that corrections, enhancements or adaptations to the validated software sustain the required software safety integrity level and follow the modification procedures.
- *Software Verification:* The activity is used, to the extent required by the safety integrity level, to test and evaluate the outputs from a given software safety lifecycle activity to ensure correctness and consistency with respect the standards and the provided inputs.
- *Software Functional Safety Assessment:* The purpose of this activity is to investigate and arrive at a judgement on the functional safety achieved by the system.

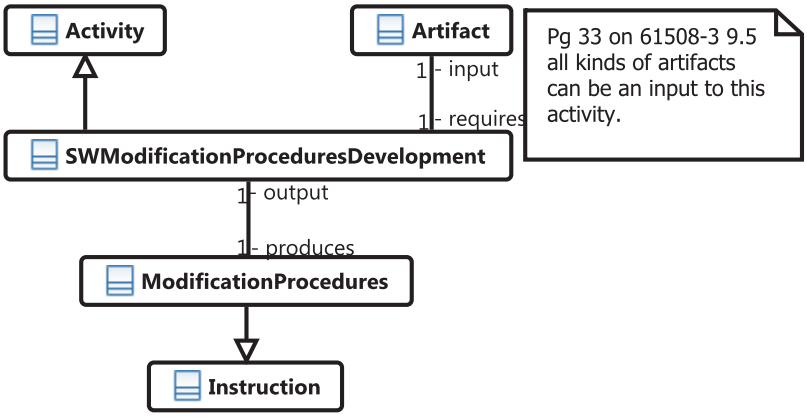


Figure 46: Software Modification Procedures Development

2 Domain Model of a Sub-Sea Production System

In Figure 49, we present a high-level domain model of a sub-sea production system. This was the model created during the case study described in Section 5.1.5

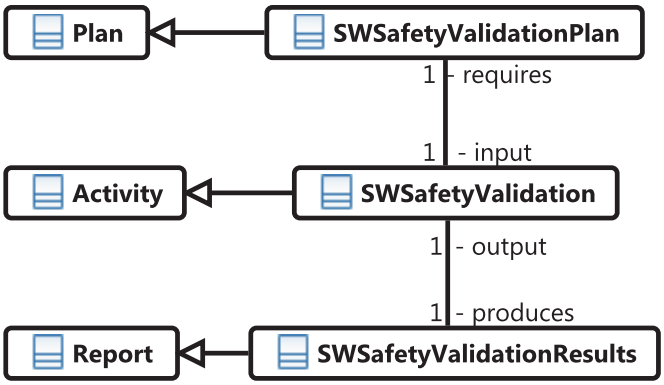


Figure 47: Software Safety Validation

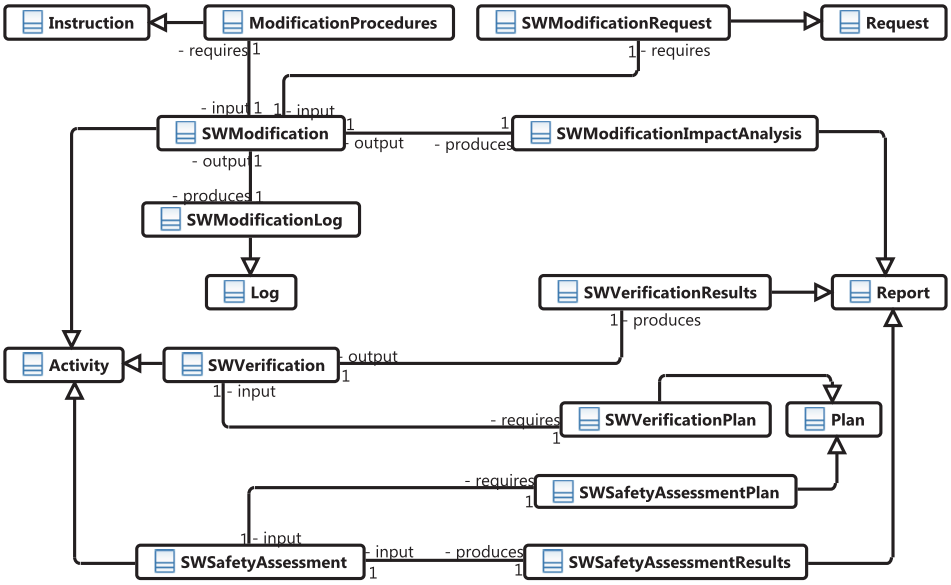


Figure 48: Software Modification, Verification, and Functional Safety Assessment

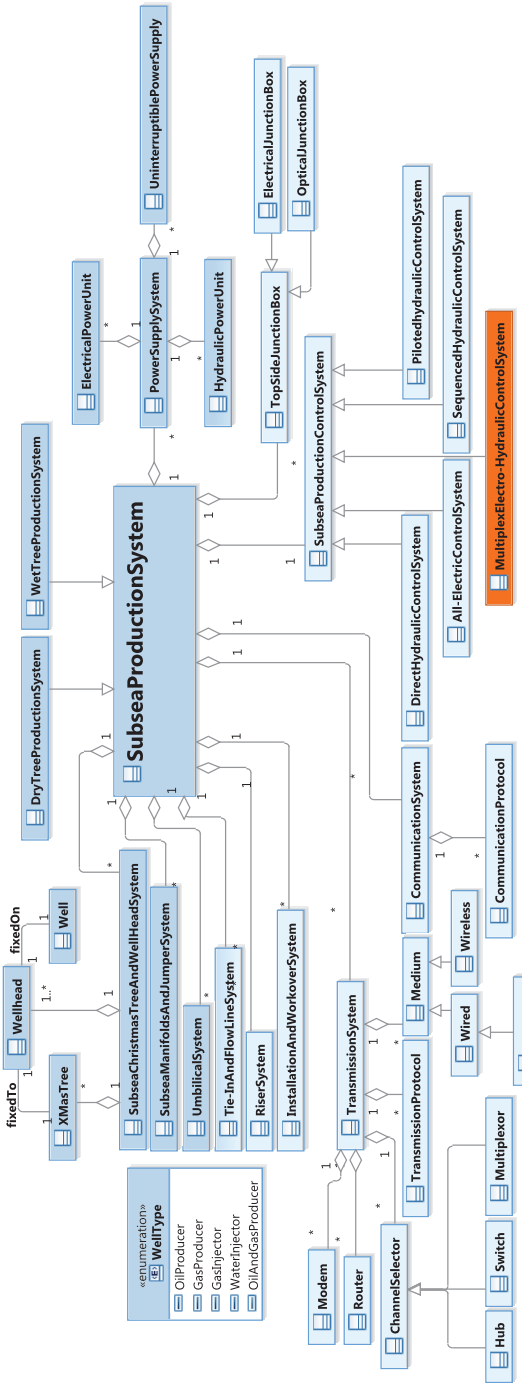


Figure 49: A High-level Domain Model of a Sub-sea Production System

Bibliography

[1] *BIRT Business Intelligence and Reporting Tools, key = Eclipse Foundation, howpublished =* `http://www.eclipse.org/birt/phoenix/`.

[2] *OMG Object Constraint Language*.

[3] Y. BAI AND Q. BAI, *Subsea Engineering Handbook*, Elsevier, 2010.

[4] K. BERKENKÖTTER, *Ocl-based validation of a railway domain profile*, in MoDELS Workshops, 2006, pp. 159–168.

[5] K. BERKENKÖTTER AND U. HANNEMANN, *Modeling the railway control domain rigorously with a uml 2.0 profile*, in SAFECOMP, 2006, pp. 398–411.

[6] P. CHUNG, L. CHEUNG, AND C. MACHIN, *Compliance flow - managing the compliance of dynamic and complex processes*, Knowledge-Based Systems, 21 (2008), pp. 332–354.

[7] T. COCKRAM AND B. LOCKWOOD, *Electronic safety cases: Challenges and opportunities*, in Current Issues in Safety-Critical Systems, in proceedings of Safety Critical Systems Symposium, Springer, 2003.

[8] I. DEY, *Qualitative data analysis - A user-friendly guide for social scientists*, Routledge, 1993.

[9] W. R. DUNN, *Practical Design of Safety-Critical Computer Systems*, Reliability Press, 2002.

[10] R. FELDT, R. TORKAR, E. AHMAD, AND B. RAZA, *Challenges with software verification and validation activities in the space industry*, in ICST'10, 2010, pp. 225–234.

[11] M. HUHN AND H. HUNGAR, *Uml for software safety and certification*, in Model-Based Engineering of Embedded Real-Time Systems, H. Giese, G. Karsai, E. Lee, B. Rumpe, and B. SchÃd'tz, eds., vol. 6100 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2011, pp. 201–237.

[12] *IBM Rational Software Architect*. `http://www.ibm.com/developerworks/rational/products/rsa/`.

[13] *Railway Applications âĂŞ Safety-related electronic railway control and protection systems.*, 1999.

[14] *Functional safety - safety instrumented systems for the process industry sector (IEC 61511).*, 2003.

[15] *Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508)*, 2005.

[16] *Petroleum and natural gas industries - design and operation of subsea production systems (ISO 13628)*, 2005.

[17] *Road vehicles – functional safety*, 2009. ISO draft standard.

[18] C. LARMAN, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)*, Prentice Hall, Oct. 2004.

[19] R. LEWIS, *Safety case development as an information modelling problem*, in Safety-Critical Systems: Problems, Process and Practice, Springer London, 2009, pp. 183–193.

[20] M. B. MILES AND A. M. HUBERMAN, *Qualitative data analysis: an expanded sourcebook*, SAGE, 1994.

[21] J. MYLOPOULOS, *Conceptual modelling and telos*, in Conceptual Modelling, Databases and CASE: An Integrated View of Information Systems Development, P. Loucopoulos and R. Zicari, eds., Wiley, thirteenth ed., 1992, pp. 49âĂŞ–68.

[22] *Safety and automation system (SAS)*, 2001.

[23] B. OATES, *Researching information systems and computing*, SAGE, 2006.

[24] *UML 2.0 Superstructure Specification*, August 2005.

[25] *UML profile for modeling quality of service and fault tolerance characteristics and mechanisms specification*. http://www.omg.org/spec/QFTP/1.1/, 2008.

[26] *UML profile for modeling and analysis of real-time and embedded systems (marte)*. http://www.omg.org/spec/MARTE/1.0/, 2009.

[27] *Software Assurance Evidence Metamodel (SAEM)*. http://www.omg.org/spec/SAEM/, 2010.

[28] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *A model-driven engineering approach to support the verification of compliance to safety standards*, in ISSRE, 2011, pp. 30–39.

[29] R. K. PANESAR-WALAWEGE, M. SABETZADEH, AND L. BRIAND, *Using uml profiles for sector-specific tailoring of safety evidence information*, in Proceedings of the 30th international conference on Conceptual modeling, ER'11, Springer-Verlag, 2011, pp. 362–378.

[30] R. K. PANESAR-WALAWEGE, M. SABETZADEH, L. BRIAND, AND T. COQ, *Characterizing the chain of evidence for software safety cases: A conceptual model based on the iec 61508 standard*, in Proceedings of the 2010 Third International Conference on Software Testing, Verification and Validation, IEEE Computer Society, 2010, pp. 335–344.

[31] *DO-178B: Software considerations in airborne systems and equipment certification*, 1982.

[32] F. REDMILL, *Installing IEC 61508 and supporting its users – nine necessities.*, in 5th Australian Workshop on Safety Critical Systems and Software, 2000.

[33] E. ROGERS, *Diffusion of innovations, Fifth Edition*, Free Press, New York, 2003.

[34] P. RUNESON AND M. HÖST, *Guidelines for conducting and reporting case study research in software engineering*, Empirical Softw. Engg., 14 (2009), pp. 131–164.

[35] N. SANNIER, B. BAUDRY, AND T. NGUYEN, *Formalizing standards and regulations variability in longlife projects. a challenge for model-driven engineering.*, in MoDRE workshop, 2011, pp. 225–234.

[36] *Application of IEC61508 and IEC61511 in the Norwegian Petroleum Industry*, 2004.

[37] G. ZOUGHBI, L. BRIAND, AND Y. LABICHE, *Modeling safety and airworthiness (RTCA DO-178B) information: conceptual model and uml profile*, Software and Systems Modeling, (2010), pp. 1–31.