Model Management for Regulatory Compliance: a position paper

Sahar Kokaly
McMaster Centre for Software Certification
McMaster University
Hamilton, Canada
kokalys@mcmaster.ca

Rick Salay
Department of Computer Science
University of Toronto
Toronto, Canada
rsalay@cs.toronto.edu

Mehrdad Sabetzadeh
SnT Centre for Security, Reliability and Trust
University of Luxembourg
Luxembourg
mehrdad.sabetzadeh@uni.lu

Marsha Chechik
Department of Computer Science
University of Toronto
Toronto, Canada
chechik@cs.toronto.edu

Tom Maibaum
McMaster Centre for Software Certification
McMaster University
Hamilton, Canada
maibaum@mcmaster.ca

ABSTRACT

Software has come to mediate many of the activities in life, including financial service platforms, social networks and vehicle control. As a result, governing bodies have responded to this trend by creating standards and regulations to address issues such as safety and privacy. In this context, the compliance of software development to standards and regulations has emerged as a key issue. For software development organizations, compliance is a complex and costly goal to achieve. They may have to comply with multiple standards due to multiple jurisdictions or to address different aspects of the software and these may overlap and conflict with each other. The evolution of standards must be tracked and changes assessed. Evidence for claims of compliance must be collected and managed. Finally, maintaining families of related software products (product lines) further multiplies the effort. In this paper, we propose to exploit the connection between the field of model management and the problem of compliance management and explore how to use model management techniques to address software compliance management issues.

Keywords

Standards, model management, regulatory compliance, assurance cases, certification.

1. INTRODUCTION

From vehicle control to social networks to business transaction platforms, software has come to mediate much of life's activities. In order to protect the best interests of citizens, responsbile organizations (e.g., International Organization for Standardization (ISO)) have responded to this trend by creating standards to address issues such as safety, security and privacy. In this environment, compliance of software to standards and regulations has emerged as a key issue. For organizations, compliance is a complex and costly goal to achieve. They may have to comply with multiple standards due to multiple jurisdictions or to address different aspects of the software which may overlap or conflict. The evolution of standards must be tracked and changes assessed. Evidence for claims of compliance must be collected and managed. Finally, maintaining families of related software products further multiplies the effort. Increasingly, models and model-driven engineering are being used as means to facilitate communication and collaboration between the stakeholders in the compliance value chain and further to introduce automation into regulatory compliance tasks. A complexity problem also exists with the proliferation of software models in model-based software development, and the field of Model Management (MM) has emerged to address this challenge. MM focuses on a high-level view in which entire models and their relationships (i.e., mappings between models) can be manipulated using specialized operators to achieve useful outcomes. We propose to exploit this connection between model driven engineering and regulatory compliance, and explore how to use MM techniques to address software compliance management issues.

Specifically, we wish to set out a research agenda that includes the following:

1. Using model management as a level of abstraction to structure and manage the complexity of the compliance problem.

2. Using model relationships to express traceability between artifacts used for compliance checking.

3. Using model management operators to (semi-)automate compliance management activities such as standard and artifact evolution or extracting relevant portions of standards.

Although the ideas put forward in this position paper are general, for illustration purposes, we ground our discussion on safety certification and refer to the ISO 26262 standard [16], which addresses functional safety of road vehicles. The

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MiSE'16, May 16-17, 2016, Austin, TX, USA

© 2016 ACM. ISBN 978-1-4503-4164-6/16/05…$15.00

DOI: http://dx.doi.org/10.1145/2896982.2896985

Table 1: Summary of Compliance Management Problems related to Model Management.

Problem	Description
P1	Can we provide a general model of compliance that can be used to define explicit relationships between the various artifacts of the compliance activity?
P2	Can evidence be reused due to standard or product evolution?
P3	Can we extract relevant parts of a standard or a system for checking compliance?
P4	Can we reduce the effort of complying to multiple standards?
P5	Can we define a way to lift compliance from single products to product lines?
P6	Can we identify relationships between standards?

rest of this paper is structured as follows: In Section 2, we give a compliance example and introduce the problems we want to use model management to address. In Section 3, we present background material on model management. In Section 4, we discuss related work. In Section 5, we address the problems stated in the example using the techniques from model management. Finally, in Section 6, we discuss the research questions that arise from applying model management to compliance and conclude with next steps.

2. MOTIVATING EXAMPLE

This section presents an example which we use to motivate the use of model management in the area of regulatory compliance. We present some specific scenarios and generalize them as problems **P1**-**P6**.

Consider an automotive company SafeCar that produces vehicles. Now suppose that SafeCar either needs to, or would like to, do the following: (1) check the compliance of its braking system safety management lifecycle with the ISO 26262 standard [16] and (2) ensure that its braking system can be certified as being safe. Typically, standards like ISO 26262 are large and take the form of textual documents, and users of the standards often face difficulty answering the questions above [19]. ISO 26262 spans 10 parts, including more than 750 clauses, and totaling approximately 450 pages. The size alone makes it difficult to comprehend, thus hindering its application and evaluation. Providing structure to the compliance problem and reducing the complexity and effort of working with standards would provide cost-benefits to these users. Moreover, when dealing with multiple standards and multiple products, SafeCar needs a mechanism to ensure explicit traceability between the various standards and between its products and the standards (**P1: Can we provide a general model of compliance that can be used to define explicit relationships between the various artifacts of the compliance activity?**).

Fig. 1 illustrates at a high level the relationships between the various artifacts involved in the compliance activity in our motivating example. The IEC 61508 standard regulates general functional safety of electrical/electronic safety related systems. ISO 26262 is an adaptation of IEC 61508. We show what a conceptual model of part of these standards may look like, in this case particularly relating to the activity of Hazard Analysis within a development lifecycle. SafeCar will engineer its software development lifecycle to be constrained by the ISO 26262 definition of a software development lifecycle. SafeCar may be interested in demonstrating that its general development process indeed complies with ISO 26262. Moreover, if SafeCar wants to achieve certification that its braking system is safe, its braking system development process will have to identify a set of safety requirements that need to be met. The safety requirements are defined in such a way as to prevent hazards (potential sources of harm caused by malfunctioning behaviour [16]). Safety cases are built to demonstrate that the product meets the safety requirements identified. A safety case may take the form of a GSN Safety Argument [17] as shown in Fig. 1[1].

But compliance is not necessarily a one-time type of activity; if SafeCar knows that a product complies with ISO 26262, and ISO 26262 undergoes a revision, SafeCar would like to have a means of checking compliance to the new version of ISO 26262 without starting from scratch. Ideally, they would like to reuse as much of the compliance artifacts (arguments, claims and evidence) used to demonstrate compliance to the older version of the standard as possible. Similarly, if the product they have already certified itself evolves to a newer version, they would like to reuse as much of the compliance artifacts from the previous version as possible (**P2: Can evidence be reused due to standard or product evolution?**). In some cases, a company like SafeCar may choose to demonstrate partial compliance to a standard; this could be due to business decisions to minimize the cost of compliance, or perhaps to address only a part of the standard that is of interest (e.g., Hazard Analysis and Risk Assessment). Similarly, the company may choose to address compliance of a part of its system to a standard (e.g., only parts that interact with the braking system.) (**P3: Can we extract relevant parts of a standard or a system for checking compliance?**).

Furthermore, SafeCar may be interested in checking that a product (e.g., its infotainment system that runs on its vehicles) complies with multiple standards (e.g., a privacy standard and a security standard) and would like to achieve this in a one-step compliance check (**P4: Can we reduce the effort of complying to multiple standards?**).

As products at SafeCar may belong to a product line, SafeCar may be interested in ensuring that the entire product line complies with a standard (**P5: Can we define a way to lift compliance from single products to product lines?**).

On the other hand, regulatory bodies may be interested in comparing their standards (e.g., ISO 26262 for automotive) with standards in a different domain (e.g., RTCA DO-178B for avionics) to understand the commonalities and the differences (**P6: Can we identify relationships between standards?**).

The above problems are summarized in Table 1. In the rest of the paper, we demonstrate how model management and model management operators can potentially be adapted

[1]IEC65108 Hazard concept model from [24], ISO 26262 Hazards and Risk Assessment model from [5] and GSN Safety Argument from slides based on [15].

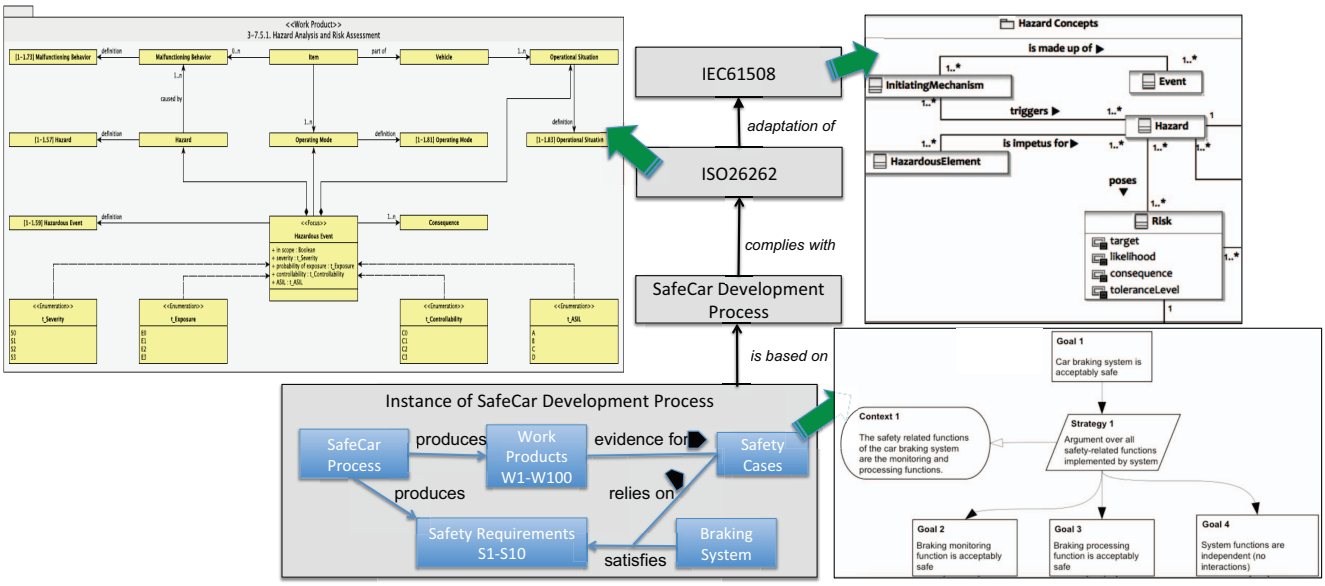

Figure 1: A Motivating example.

to help address these problems.

3. MODEL MANAGEMENT

In the field of model-driven software development [3], a complexity problem exists due to the proliferation of software models. As such, the area of Model Management [2] has emerged to address this challenge. Model management focuses on a high-level view in which entire models and their *relationships* (i.e., mappings between models) can be manipulated using *operators* (i.e., specialized model transformations) to achieve useful outcomes. To help visualize and manipulate collections of models and their relationships, model management uses a special type of model called a *megamodel* [8] in which the elements represent models and the links between the elements represent relationships between the models. For example, Fig. 1 is a megamodel. Model management operators that have been studied include the following:

- The *slice* [21] operator accepts a model and a *slicing criterion* and extracts the subset of the model satisfying the criterion. Model slicing is a way to manage model complexity by focusing on a relevant subset of a model.
- The *match* [2] operator accepts two models and produces a relationship containing mappings between equivalent (or similar) elements in the models. This is usually interpreted as identifying the overlap between the models.
- The *diff* [2] operator accepts two models and produces a model that represents the differences between the models. Model differencing aids the comparison of model content, e.g., across different versions.
- The *merge* [4] operator accepts two models and a relationship expressing the overlap between them and produces a model that combines the content of the models according to the overlap. Model merge must address the issue of conflicts that could occur when the content is combined.
- The *lift* [25] operator accepts a model transformation and produces a product line transformation that behaves the same way as the original model transformation for each product in the product line. Transformation lifting saves effort by allowing model transformation to be reused for product lines of models.
- The *filter* [26] operator accepts a megamodel and a model (relationship) property and produces a megamodel with all models (relationships) not satisfying the property removed. Filtering a megamodel is useful for managing the complexity of large collections of related models.
- The *map* [26] operator accepts a megamodel and a model transformation to produce the megamodel that results from applying the transformation to all applicable models and relationships in the input megamodel. Mapping a transformation over a megamodel is used to reduce the effort of applying a transformation to the elements of a large collection of related models.

Each of these operators can be viewed as an *abstract* transformation that defines a class of concrete transformations - i.e, the implementations that refine the operator for particular model types. For example, a model merge of class diagrams is implemented differently than a model merge of state machines. Another widely used class of transformations used in model management are *bidirectional transformations* [9]. Bidirectional transformations are used to keep two related models synchronized when one of the models changes (e.g., via model co-evolution, correction, etc.) by generating the update for the other model.

4. RELATED WORK

We know of no existing research specifically on applying model management techniques to compliance management problems; however, there is substantial research that is related to using model-based approaches to aid in the compliance problem. We consider this research to be complementary to our objectives and review it here.

Compliance management frameworks have been proposed. [14] sketches a proposal for comprehensive compliance management in software organizations. Although this is at a higher-level of abstraction than our proposal, the authors discuss the need for tool support for working on large, possibly overlapping or conflicting compliance documents – this is clearly within the scope of our proposal. Researchers have proposed generic metamodels that can be used to structure

any safety standard as well as the related safety case information in a project [6, 13]. An advantage of such an approach would be the possibility of providing generic tooling to address the compliance to any safety standard. Thus, we consider these proposals to be complementary to our objectives and will investigate their feasibility as the basis for model management operators.

Specific algorithms for different aspects of compliance management have been proposed. For example, Nejati et al. [21] have developed a model slicing algorithm for extracting parts of a design that are relevant to a given safety requirement; the survey [11] identifies research strands proposing methods for the extraction of requirements models from regulation documents; and, the authors of [12] propose an algorithm for comparing multiple regulations. We view these strands as complementary to our proposal as they can form the basis for the implementation of model management operators.

Argumentation modeling languages have been studied extensively as the basis for expressing compliance claims. These include the Goal Structured Notation(GSN) [17], Claim, Arguments and Evidence (CAE) [10] and OMG's Structured Assurance Case Metamodel (SACM) [2]. Of these, SACM represents the latest in the evolution of these notations and is also proposed as a standard. We aim to use modeling languages such as these to express the compliance relationships used in model management of compliance.

Standards and regulations can be expressed as models. For example, [18] shows that the ISO 26262 standard for functional safety of road vehicles can be represented by a combination of a structure model, conceptual model, process model while [24] proposes a conceptual model of IEC 61508. We consider this work as a prerequisite for applying model management techniques to the compliance problem.

A recent survey [20] regarding how industry addresses evidence management for compliance to safety standards, reveals that much of evidence management is done manually with little reliance on advanced tool support. However, key issues such as evidence traceability, structuring evidence as models and assessing evidence completeness or correctness correspond directly to general modeling issues: expressing model relationships (including traceability relationships), structuring model content using metamodels and checking model completeness/correctness. We may conclude that adapting automated or semi-automated techniques from model management to address these issues can benefit the state of the practice.

5. MODEL MANAGEMENT FOR COMPLIANCE

Since standards and regulations can be expressed as models (e.g., [18]), and compliance claims and links to evidence can also be captured in a model [11], the modeling community is starting to work in the direction of providing support for compliance management. Specifically, the Object Management Group has recently issued a modelling language standard called the Structured Assurance Case Metamodel (SACM) [22].

There is evidence that the MM approach to compliance management may fill the gaps left by other research on this topic. For example, a recent survey [1] comparing international research in compliance management with the compliance needs of the Australian industry found that a key area where support is needed is in managing the impact of regulations. Specifically, they identified four factors: (1) frequent changes to regulations; (2) legislation weaknesses; (3) inconsistencies; and, (4) overlap in regulations. Furthermore, they identified a disproportionate lack of research in these areas as compared to other issues in compliance management. Although they hypothesized that this may be due to the fact that these factors may be in the legal (as opposed to IT) realm, all but factor (2) correspond to general modelling problems that have been addressed with model management techniques. For example, (1) corresponds to the change propagation due to model evolution, (3) to addressing inconsistencies, which is a standard step within a model merge operation, and (4) to identifying overlaps, which is the outcome of the model match operation.

In this section, we take advantage of the above, and demonstrate our ideas for using MM techniques in the area of compliance. We refer to the problems identified in our motivating example, which are also summarized in Table 1, and propose general solutions to them.

P1: Creating a general model of compliance that defines explicit relationships between the compliance artifacts. We illustrate our general model of compliance in Fig. 2. The idea is that regulators (such as the ISO organization) define standards (such as ISO 26262), and these standards are defined as a collection of interrelated models. Each model addresses a different view (process, concepts, etc.) of the standard. In model management terminology, a megamodel [8] represents a model of models and relationships between them. A standard can then be seen as a megamodel in this sense, but for simplicity, we illustrate it as a single model and refer to it as the Standard Model (SM). A company would ideally also have a model of its development process, which again can be seen as a megamodel linking different views of the company's software development process. Again, for simplicity, we will illustrate it as a single model and refer to it as the Software Development Process Model (SDPM). The SDPM should be engineered to satisfy constraints in the SM; therefore, compliance with the SM can be shown via an Assurance Case (AC) that is defined as a relationship between SM and SDPM.

The SDPM can undergo refinements within the company and can be tailored with respect to the product it will be used in producing. However, as soon as the company creates an instance of its SDPM for a certain product, this instance should conform to the metamodel defined by the SDPM. For example, concepts and processes in the instance should conform to their respective definitions in the SDPM. Depending on the property "P" that is to be met (Safety, Privacy, Security, etc.), "P"-Requirements are produced as part of the instance process, along with the various work products that are defined in the standard. When preparing to certify an artifact to meet the property "P", assurance cases (or "P"-Cases) are provided to support this. The Assurance Case (AC) is constructed using evidence (given by the work products), and can be seen as an instance of a GSN [17] or an OMG SACM [22] metamodel which defines how to construct these instances.

Fig. 2 depicts this model via two levels of compliance. The top-level (or *process*) compliance is given by an assurance case, either stating that an item complies or providing rationale for non-compliance. The bottom-level (or *product*)

[2]OMG: http://www.omg.org/spec/SACM/1.1/

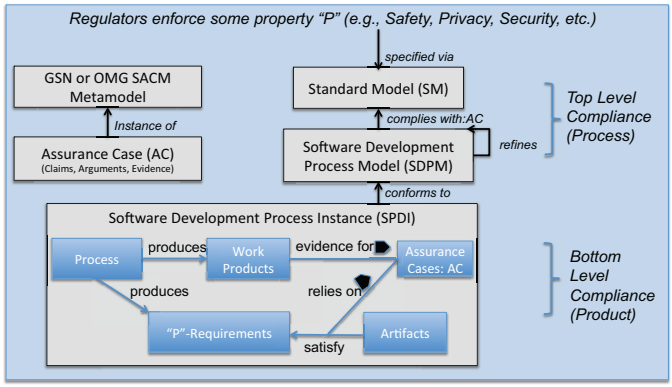

Figure 2: A general model of compliance.

compliance is given by a "P"-case that demonstrates that the product satisfies the property "P" as specified in the standard. In the UML terminology, Fig. 1, which describes a concrete example, can be seen as an Object Diagram that is an instance of the Class Diagram given by our general model in Fig. 2.

This general model of compliance need not be limited to a single standard and a single product. For example, Fig. 3 shows a model with several products – a car's braking system and its infotainment system – complying to multiple standards – safety, security and privacy.

Note that the model in Fig. 2 is itself a megamodel: the models represent the standards, processes and artifacts, and the relationships between them are made explicit. Some examples of these relationships are: *specialization* (of a standard from a general domain to a more specific one), *compliance* (of a process to a standard), *refinement* or *evolution* (of a standard) and *instantiation* (as in the case of the instance of the SDPM). This observation opens up the opportunity of using collection-based operators [26] on megamodels of standards. One possible scenario is: given a collection of standards, use the *filter* operator to obtain "relevant ones" based on some criteria. Perhaps we are interested in the process-related standards, so we filter out all the ones with type "activity diagram". Also, given a set of standards with a shared ontology, we can use the *map* operator to translate some concept in the ontology to another (e.g., "automobile" to "vehicle").

P2: Reusing evidence due to standard or product evolution. Evolution of a standard involves a transformation of the standard model from an old version to a newer one. But since there might have been some products that were already shown to comply with the standard, its evolution means that the compliance of these products needs to be re-evaluated. More concretely, in Fig. 1, suppose the ISO 26262 standard undergoes a revision (which it currently does), leading to the introduction of a new attribute, say "cost", in the "Risk" class. This means that the SafeCar Development Process has to evolve to ensure that it complies with the newer version of the standard. This problem is commonly addressed in model management using a *bidirectional model transformation* as discussed in Section 3. The main idea is that a *match* operator is used to establish links between the old and new versions of ISO 26262, and a *diff* operator is used to show what new information has been added. Then, the SafeCar Development Process is transformed to a new version that is compliant with the new version of ISO 26262. The challenge added here is in managing the assurance case artifacts (claims, arguments and evidence) that are attached to the compliance relationship. Not only do these need to be re-evaluated, but it would be very cost-effective if these artifacts could be reused as much as possible. Evolution of a product (e.g., the braking system in Fig. 1) that is known to comply with a standard also means that compliance needs to be re-evaluated. Again, this can be expressed via a *bidirectional model transformation* definition which also needs to be adapted to take into account the assurance case artifacts and reuse them as much as possible to minimize the cost of compliance.

P3: Extracting relevant parts of a standard or a system for checking compliance. Based on some criteria of interest (*slicing criteria*), extracting only relevant parts of a standard (e.g., those related to hazards and risks in ISO 26262) can be achieved with the use of the *slice* operator. This could also apply to the product undergoing compliance; the slicing criteria could be chosen to select parts of the model related to the braking system for example. Both scenarios (standard and product slicing) could arise due to business decisions to demonstrate partial compliance with a standard in order to narrow the focus to the compliance tasks that are feasible to perform given the stage of development that the system is at.

P4: Effort reduction of compliance to multiple standards. As demonstrated in Fig. 3, we may wish to check the compliance of a system (e.g., infotainment) to multiple standards (e.g., security and privacy). Some concepts presented in these standards may overlap (e.g., the notion of a "threat"). An MM operator *match* can be used to help identify these overlaps. Once their overlap is defined (note that this can be empty), standards can be merged with the *merge* operator creating a merged standard with traceability mappings back to the original standards. These mappings are important to have, especially when the original standards evolve, which can be reflected in the merged standard (via model co-evolution). Compliance of the infotaintment system in Fig. 3 to the security and privacy standards can therefore be reduced to a single compliance checking problem of the system to the merged standard. As in model management, a challenge here is actually creating the overlaps between the standards which involves a high degree of human input. Another challenge is understanding how to provide assurance case artifacts when demonstrating compliance to the merged standard (e.g., how do these artifacts relate to the artifacts we would get when demonstrating compliance to each of the individual standards.).

P5: Lifting compliance from a single product to a product line. Products are often modified and reused in new applications, forcing companies to develop and maintain product lines. Product line safety is a natural important requirement of this activity [23]. Lifting the problem of compliance of a single piece of software to a single standard to that of an entire software product line can be achieved using the model management *lift* operator. Once again, the challenge here is understanding how the assurance artifacts are expressed in the lifted version of the compliance relationship.

P6: Identifying relationships between standards. In order to create a common certification framework which spans different vertical markets, e.g., in the transport sec-

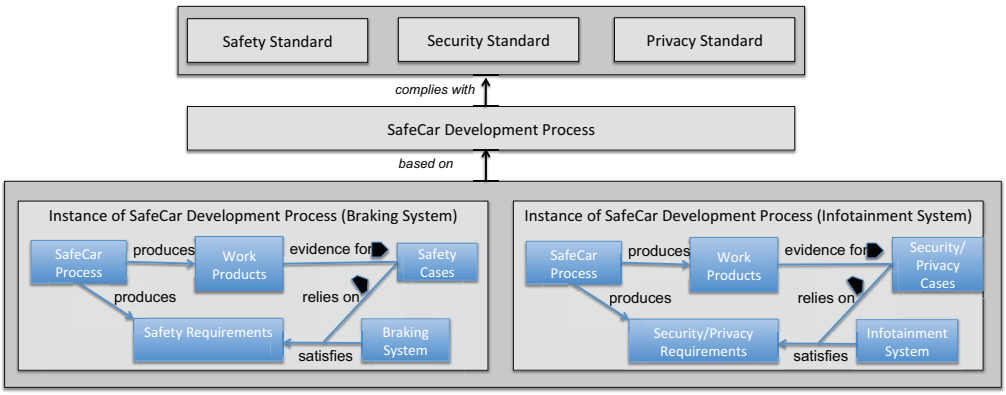

Figure 3: An example of compliance of multiple artifacts to multiple standards.

tor (railway, avionics and automotive industries), and facilitates the reuse of assurance assets within, across, and between domains, creating mappings between these standards is an important step. The *match* operator can be used to identify these overlaps, in the simplest case using a "name-match" criterion which links elements with matching names. Creating these overlaps is also an essential step in performing other useful operations such as *merge* and for maintaining consistency between standards (model synchronization). Another relationship that one may be interested in identifying is representing the differences between two standards (these could be two versions of the same standard). This could be seen as differences in concepts (e.g., what appears in one and not in the other), and can be achieved using the model management *diff* operator.

Companies like SafeCar could benefit from the automation of parts of the compliance activities. For example, we have identified the following in Part 2 of the ISO 26262 standard:

"The Safety Case has to be reviewed for completeness:
C.2 Review of the completeness of the safety case (see 6.5.3)
C.2.1 Confirmation that the work products referenced in the safety case are available and sufficiently complete, so that the item's achievement of functional safety can be adequately evaluated.
C.2.2 Confirmation that the work products referenced in the safety case: are traceable from one to another, have no contradictions within or between work products, and either have no open issues that can lead to the violation of a safety goal, or have only open issues that are controlled and have a plan for closure."

Once the standards and compliance artifacts are structured in a model management framework, various techniques can be used to analyze and verify these models. In the ISO 26262 example, activities like checking the completeness of a safety case, checking the availability (existence) and the completeness of work products, checking traceability between work products, checking for contradictions (conflicts) can certainly be at least semi-automated given appropriate models that are supported by expert judgement.

Another general advantage of using a model management framework for managing standards and compliance artifacts is the use of generic model transformations. These could be useful for a variety of reasons: 1. Providing different views (models) of the same standard which could aid in improving communication and understandability. 2. Transformation of standards if companies were to adopt regulations and best practices from a different country/jurisdiction .

6. SUMMARY AND NEXT STEPS

This work should not be seen as a direct application of model management to classical meta-data problems, particularly due to the challenges introduced when applying it in the context of regulatory compliance. These challenges appear due to three factors: 1. The amount of natural language used in expressing the standards and the assurance cases. 2. The human-in-the-loop factor and reliance on expert opinion. 3. The assurance artifacts that need to be managed when applying the various model management operators.

Next, we summarize our research agenda for exploring the use of model management to support the activity of compliance management by presenting a set of research questions.

6.1 Research Questions

RQ1: What types of relationships exist between standards, software development processes and artifacts (e.g., mappings, refinements, etc.)? We have identified some in this paper, but are they sufficient for representing a complete compliance management framework?

RQ2: What is a reasonable and complete general model of compliance that takes into account the various relationships identified in RQ1? A challenge here is that some of these relationships are not standard model management relationships since they encapsulate assurance cases which in turn contain claims, arguments and evidence. These assurance cases have to be modelled and incorporated in the compliance relationships.

RQ3: What type of traditional model management operators can be used in the context of compliance management? We have identified some in this paper, but are there others? Also, given these operators, how can they be defined to adapt to compliance management (e.g., how do they handle the various assurance case artifacts)?

RQ4: Since compliance can be measured by a degree (i.e., it is not a binary relation), this differs from usual binary relationships between models. How can we adapt model relationships and model management operators to address this?

RQ5: Automation can be seen as an advantage brought by model management. As compliance management is a highly human-in-the-loop activity due to the need of expert judgement to ensure compliance and create assurance cases, can we identify parts of the compliance management problems that can be semi-automated, and what kind of tool support

(modeling, model checking, code generation) can be provided to help with semi-automating them? Furthermore, when the automated parts of the process create partial (i.e., incomplete) artifacts, how can we efficiently integrate human intervention to complete the artifacts?

6.2 Next Steps

Our next steps will be addressing the research questions identified in this paper. We will focus primarily on demonstrating the advantages model management can bring to the area of regulatory compliance in terms of re-usability of the compliance artifacts and supporting multiple standards and product lines. Also, given a model management tool such as MMINT [7], we would like to implement the adapted model management operations described in the paper and use it as a tool for structuring and managing compliance artifacts. Ideally, we would like to conduct a case study with our industrial partner GM within the NECSIS project, where we would specifically consider the ISO 26262 [16] standard for functional safety of road vehicles to help understand the applicability and limits of our approach. We are currently collaborating on providing a conceptual model of ISO 26262 [5].

7. ACKNOWLEDGMENTS

This work is being done as part of the NECSIS project (www.necsis.ca), funded by Automotive Partnership Canada and NSERC.

8. REFERENCES

[1] N. S. Abdullah, S. Sadiq, and M. Indulska. Emerging Challenges in Information Systems Research for Regulatory Compliance Management. In *Proc. of CAiSE'10*, pages 251–265. Springer, 2010.

[2] P. A. Bernstein. Applying Model Management to Classical Meta Data Problems. In *Proc. of CIDR'03*, volume 2003, pages 209–220, 2003.

[3] S. Beydeda, M. Book, V. Gruhn, et al. *Model-driven software development*, volume 15. Springer, 2005.

[4] G. Brunet, M. Chechik, S. Easterbrook, S. Nejati, N. Niu, and M. Sabetzadeh. A Manifesto for Model Merging. In *Proc. of GAMMA@ICSE'06*, pages 5–12. ACM, 2006.

[5] V. Cassano, N. Singh, S. Grigorova, L. Patcas, S. Kokaly, M. Lawford, T. Maibaum, and A. Wassyng. Making Sense of ISO 26262: Some Structured Views, 2016. Submitted to Reliability Engineering and System Safety journal.

[6] J. L. de la Vara and R. K. Panesar-Walawege. Safetymet: A Metamodel for Safety Standards. In *Proc. of MODELS'13*, pages 69–86. Springer, 2013.

[7] A. Di Sandro, R. Salay, M. Famelis, S. Kokaly, and M. Chechik. MMINT: A Graphical Tool for Interactive Model Management. In *Proc. of MODELS'15 (demo track)*, 2015.

[8] Z. Diskin, S. Kokaly, and T. Maibaum. Mapping-Aware Megamodeling: Design Patterns and Laws. In *Proc. of SLE'13*, pages 322–343, 2013.

[9] Z. Diskin, Y. Xiong, and K. Czarnecki. From state-to delta-based bidirectional model transformations. In *Theory and Practice of Model Transformations*, pages 61–76. Springer, 2010.

[10] L. Emmet and G. Cleland. Graphical Notations, Narratives and Persuasion: A Pliant Systems Approach to Hypertext Tool Design. In *Proc. of Hypertext'02*, pages 55–64. ACM, 2002.

[11] S. Ghanavati, D. Amyot, and L. Peyton. A Systematic Review of Goal-Oriented Requirements Management Frameworks for Business Process Compliance. In *Proc. of RELAW'11*, pages 25–34. IEEE, 2011.

[12] S. Ghanavati, A. Rifaut, E. Dubois, and D. Amyot. Goal-Oriented Compliance with Multiple Regulations. In *Proc. of RE'14*, pages 73–82. IEEE, 2014.

[13] I. Habli and T. Kelly. A Model-Driven Approach to Assuring Process Reliability. In *Proc. of ISSRE'08*, pages 7–16. IEEE, 2008.

[14] A. Hamou-Lhadj and A. Hamou-Lhadj. Towards a Compliance Support Framework for Global Software Companies. In *Proc. of SEA'07*, pages 182–192, 2007.

[15] R. Hawkins, I. Habli, D. Kolovos, R. Paige, and T. Kelly. Weaving an Assurance Case from Design: A Model-Based Approach. In *Proc. of HASE'15*, pages 110–117. IEEE, 2015.

[16] International Organization for Standardization. *ISO 2626: Road Vehicles – Functional Safety*, 2011. 1st version.

[17] T. Kelly and R. Weaver. The Goal Structuring Notation – A Safety Argument Notation. In *Proc. of DSN'04*, 2004.

[18] Y. Luo, M. van den Brand, L. Engelen, J. Favaro, M. Klabbers, and G. Sartori. Extracting Models from ISO 26262 for Reusable Safety Assurance. In *Proc. of ICSR'13*, pages 192–207. Springer, 2013.

[19] L. I. Millett, M. Thomas, D. Jackson, et al. *Software for Dependable Systems:: Sufficient Evidence?* National Academies Press, 2007.

[20] S. Nair, J. L. de la Vara, M. Sabetzadeh, and D. Falessi. Evidence Management for Compliance of Critical Systems with Safety Standards: A Survey on the State of Practice. *Information and Software Technology*, 60:1–15, 2015.

[21] S. Nejati, M. Sabetzadeh, D. Falessi, L. Briand, and T. Coq. A SysML-based Approach to Traceability Management and Design Slicing in Support of Safety Certification: Framework, Tool Support, and Case Studies. *Information and Software Technology*, 54(6):569–590, 2012.

[22] OMG. Structured Assurance Case Metamodel (SACM), 2015. http://www.omg.org/spec/SACM/.

[23] R. Palin, D. Ward, I. Habli, and R. Rivett. ISO 26262 Safety Cases: Compliance and Assurance, 2011.

[24] R. K. Panesar-Walawege, M. Sabetzadeh, and L. Briand. Supporting the verification of compliance to safety standards via model-driven engineering: Approach, tool-support and empirical validation. *Information and Software Technology*, 55(5):836–864, 2013.

[25] R. Salay, M. Famelis, J. Rubin, A. Di Sandro, and M. Chechik. Lifting Model Transformations to Product Lines. In *Proc. of ICSE'14*, pages 117–128. ACM, 2014.

[26] R. Salay, S. Kokaly, A. Di Sandro, and M. Chechik. Enriching Megamodel Management with Collection-Based Operators. In *Proc. of MODELS'15*, pages 236–245. IEEE, 2015.