Regulatory Compliance-aware System Change Management via an Ontology-based Approach*

Barbara Gallina[1][0000−0002−6952−1053], Markus Schweizer[2], and Henrik Dibowski[2]

[1] Mälardalen University, Västerås, Sweden, barbara.gallina@mdu.se
[2] Robert Bosch GmbH, Renningen, Germany, {name.surname}@de.bosch.com

Abstract. To meet customer demands, the automotive industry is characterised by high levels of variability, essential for product diversification. The ongoing transition to higher levels of automation and connectivity is witnessing an ongoing increase in regulatory requirements, which sometimes overlap or sometimes exhibit interdependence characteristics. Hence, the introduction of new features, in the context of an item increment, may make the item transit to a more demanding regulatory space. Typically, the impact of the change is handled manually. No automated system change management, aware of regulatory compliance, is available. In our previous work, we proposed an ontology-based representation for managing product variability and re-configuration. We briefly illustrated its usage by focusing on the variability of regulatory requirements due to different jurisdictions and their impact on the product. In this paper, we expand and use our ontology-based representation to realise an automated regulatory compliance-aware system change management. Each system change can be validated by constraints and, as a consequence, variability manager may more easily detect missing evidence that is required for compliance. Via rules, missing but required features can be automatically generated and connected. Specifically, we illustrate the usage of our ontology-based representation by considering a product increment within the product line of power-operated window lifters. The increment, enabling remote closing and opening, calls for compliance with regulations that are related not only to safety, but also to cybersecurity.

Keywords: Cybersecurity and Safety Interplay · Change Impact Analysis · Ontologies · Reusable Validation Constraints · SHACL.

1 Introduction

To meet customer demands, the automotive industry is characterised by high levels of variability, essential for product diversification [13]. The ongoing transition to higher levels of automation and connectivity is witnessing, on the one hand, an ongoing increase of regulatory requirements (e.g., UN155 [17], UN161 [18], etc.), which sometimes overlap or sometimes exhibit interdependence characteristics,

* Partially funded by Software Center via the ∞ COMPASS [15] project.

and, on the other hand, an increase of highly configurable software-intensive and software-defined functionalities. To the increase of regulatory requirements corresponds an exacerbation of the complexity of the compliance demonstration due to, among other challenges, their traceability challenge concerning traceability of their commonality, variability and interdependence. To the increase of software-intensive and highly configurable functionalities corresponds an exacerbation of the software complexity, which as stated in [1], grows overwhelmingly year by year, and whose (commonality, variability and interdependence) traceability also represents a challenge. The increasing complexity of the regulatory space goes hand in hand with the growing complexity of the engineering space. The relationships of these spaces shall be made explicit in order to contribute to coping with the complexity of the compliance problem. What was stated by Brooks [2] in the 80's of the previous century, is still valid nowadays, i.e., "the software product is embedded in a cultural matrix of applications, laws, and machine vehicles. These all change continually, and their changes inexorably force change upon software product." Hence, the dissonance, which might be created whenever a legal requirement undergoes changes and those changes are not traced down to product (functionality/software) changes, shall be handled. Similarly, the dissonance, which might be created whenever a product undergoes a change and those changes are not traced up to the changes of the legal space, shall be handled. The introduction of new features, in the context of an item increment, for instance, not only expands the configurability of the technological space but may make the item transit to a more demanding regulatory space. Typically, the change impact analysis is handled manually. No automated system change management aware of regulatory compliance is available. More generally, the traceability between variants, within the above-mentioned matrix, is still an open challenge.

To contribute to facing such challenge, in [4], we proposed a product-line oriented extension of the Rasmussen socio-technical system where the different dimensions of the above-mentioned matrix are organised in a layered structure incorporating socio-aspects (such as legislations, standards, etc.) and technical aspects (such as the vehicles and its in-mounted items). To capture the layered structure, we also proposed an ontology-based representation for managing product variability and re-configuration [3]. We briefly illustrated its usage by focusing on variability of regulatory requirements due to different jurisdictions and their impact on the product. In this paper, we further expand our representation and we use the expanded representation to realize an automated regulatory compliance-aware system change management. A system change can make the system violate constraints and the user (e.g., variability manager, safety manager, etc.) may more easily detect missing evidence that is required for compliance. Specifically, we illustrate the usage of our extended ontology-based representation by considering a product increment within the product line of power-operated window lifters to enable remote closing and opening. The increment calls for compliance with regulations that are related not only to safety, but also to cybersecurity.

The rest of the paper is organised as follows. In Section 2, we provide essential background information. In Section 3, we extend our previously proposed ontology-based representation. In Section 4, we use our extended ontology-based representation to realise an automated regulatory compliance-aware change management. In Section 5, we briefly discuss our findings and we also explain the synergy with the SPI Manifesto. In Section 6, we briefly discuss related work. Finally, in Section 7, we present our concluding remarks.

2 Background

In this section, we recall essential information regarding the problem space (power-operated window lifters, automotive regulations, and standards) and the solution space (continuous compliance via the extended Rasmussen's socio-technical system and ontology-based variability management).

Remotely-controlled power-operated window lifters. Power-operated window lifters (WLs) are typically integrated by OEMs (Original Equipment Manufacturers) based on subsystems/components supplied by different tiers (Bosch, for instance, acts as Tier1 or Tier2 in relation to different OEMs), involved within the supply chain. The main functionality of WLs is to regulate the movement of a side window pane, as a part of the vehicle's door. It should be noted that, depending on the car, a global motor-operated window system for vehicles includes two to four window lifters, which regulate the front only or the front and rear side window panes. The movement can be triggered by the driver through the master control switch or by the passenger through the switch located on the door panel. The window motor is directly connected to the switch, which then controls the lifting or lowering of the windows. In addition to internal switches, the movement can also be triggered via an external switch located within a keyfob or a digital key making WLs remotely-controlled. In the case of a digital key, specific algorithms are needed to control the opening/closing, the authorisation, and the revocation.

Remotely-controlled WLs-related UNECE requirements. Several regulations are applicable for the engineering of remotely-controlled WLs. In what follows, we limit our attention to UN161 [18], UN155 [17], and UN21 [16]. Given the partial overlap and interdependencies among these regulations, they can be captured as a product line of mandatory and tailorable requirements.

UN161 provides uniform provisions for the protection of motor vehicles against unauthorised use and the approval of the device against unauthorised use (by means of a locking system). In what follows, we recall a subset of the provisions for digital keys, focusing on the safety requirements related to risk assessment and control. **Requirement 3.4** requires a description of the safety measures designed within the digital key revocation process to ensure safe operation of the vehicle. **Requirement 4.2.1** requires the revocation of a digital key to not result in an unsafe condition. A risk reduction analysis is required using functional safety standard such as ISO 26262 and safety of the intended functionality standard such as ISO/PAS 21448 (now ISO 21448), which docu-

ments the risk to vehicle occupants caused by revocation of a digital key and the reduction of risks resulting from the implementation of the identified risk mitigation functions or characteristics. **Requirement 5** requires the effectiveness of the system to not adversely be affected by cyber-attacks, cyber threats and vulnerabilities. The effectiveness of the security measures shall be demonstrated by compliance with UN Regulation No. 155." In addition, regarding the competence of the auditors, it is stated that they shall be competent in particular for ISO 26262-2018, and ISO/PAS 21448; and shall be able to make the necessary link with cybersecurity aspects in accordance with UN Regulation No. 155 and ISO/SAE 21434. We interpret this as an indirect endorsement of standards.

UN155 provides uniform provisions for the approval of vehicles with regards to cyber security and a cyber security management system. Regarding competence of approval authorities, **Requirement 5.3.1.a** requires appropriate cyber security skills and specific automotive risk assessments knowledge. ISO 26262-2018, ISO/PAS 21448, ISO/SAE 21434 are provided as example. We interpret this as an indirect endorsement of the mentioned standards.

UN21 provides uniform provisions for the approval of vehicles with regard to their interior fittings. UN21 does not contain any standard endorsement.

Remotely-controlled WLs-related automotive standards. Several standards are applicable for the engineering of remotely-controlled WLs. In what follows, we limit our attention to the three standards, which are mentioned within the above-recalled regulations, i.e., ISO 26262 [8], ISO 21448 [9], and ISO 21434 [10]. For each standard, we recall basic information.

ISO 26262 [8] is the functional safety (FuSa) standard for road vehicles, where functional safety is defined as absence of unreasonable risk due to hazards caused by malfunctioning behaviour of electrical/electronic systems. ISO 26262 specifies a safety life-cycle to be adopted for the development of the items (i.e., systems or arrays of systems, which implement a function at the vehicle level). The stringency of the life-cycle can be tailored based on the criticality of the hazardous events. The criticality is determined by assigning an *ASIL* (Automotive Safety Integrity Level) to the hazardous events, i.e., to the combination of *hazard* (potential source of harm) and *operational situations* (scenarios that can occur during vehicle's life). The stringency of the life-cycle can also be tailored based on the tailoring rules stated within ISO 26262. Specifically, in case of a rationale, life-cycle activities can be omitted, re-grouped, re-ordered or modified according to an organisation-specific criterion. The ISO 26262 safety life-cycle comprises phases from concept through decommissioning of the system and it is based upon a V-model. The concept phase comprises *Item definition* and *HARA* (Hazard Analysis and Risk Assessment). The item definition consists of the provision of the description of the system's functionality, interfaces, and environment. Hazard Analysis and Risk Assessment, FuSa-HARA, consists of a series of activities aimed at identifying, classifying and assessing the risk.

ISO 21448 [9] is the SOTIF (Safety Of The Intended Functionality) standard for road vehicles, where SOTIF is defined as "absence of unreasonable risk due to hazards resulting from functional insufficiencies of the intended func-

tionality or its implementation". The hazards mentioned in this definition may occur due to hazardous behaviour, initiated by triggering conditions, which include reasonably foreseeable direct misuse (i.e., usage that is not intended by the manufacturer or the service provider and, if direct, can act as a potential triggering condition, which is a condition that initiates a subsequent system reaction contributing to either a hazardous behaviour or an inability to prevent or detect and mitigate a reasonably foreseeable indirect misuse). The foreseeable direct misuse is in focus in this paper. ISO 21448 does not explicitly specify a SOTIF life-cycle. However, it is stated that SOTIF activities are expected to complement the FuSa-life-cycle. Hence, a SOTIF life-cycle is implicit. The SOTIF life-cycle starts with the specification of the functionality. SOTIF specification of the functionality may consider the item definition (work product), produced as result of the item definition activity within ISO 26262. After the specification of the functionality, SOTIF-HARA (Hazard Analysis and Risk Assessment) is conducted. Also in the context of this standard, the stringency of the SOTIF life-cycle can be tailored based on the criticality of the hazardous events. Hence, SOTIF-HARA is crucial. ISO 21448 does not include specific tailoring rules in relation to re-grouping or re-ordering. This is coherent since no specific life-cycle is explicitly stated. Even if not specifically stated, it is reasonable to assume rules for typical tailoring in terms of omission or organisation-specific specialisation in line with the practices within ISO 26262.

ISO 21434 [**10**] is the cybersecurity standard for road vehicles, where cybersecurity is defined as "condition in which assets are sufficiently protected against threat scenarios to items of road vehicles, their functions and their electrical or electronic components". ISO 21434 specifies a security life-cycle to be adopted for the development of secure assets. The stringency of the lifecycle can be tailored. In what follows, we will focus on the concept phase, which first of all requires the definition of the item. Content-wise, the work-product item definition may differ from the ISO 26262 item definition even though it may partly overlap. Based on the item definition, the TARA (Threat Assessment and Remediation Analysis) can be conducted, which includes the following activities: asset identification, threat scenario identification, impact rating, attack path analysis.

Given the tailoring capabilities, which are implicitly or explicitly included in all standards, standards can be seen as product lines of mandatory and tailorable requirements. In addition, when the glossary is shared and when the level of commonality among standards is significant, even the set of standards can be captured as a product line.

Continuous compliance via the extended Rasmussen's socio- technical system. To achieve continuous regulatory compliance and assurance, changes need to be managed. Changes may take place during the engineering as well as during the operational life of a vehicle/item. In our previous work [4], we introduced an extension of the Rasmussen socio-technical system. The extension considers product lines and not single products, where products do not only represent e.g., vehicles, but also regulations, standards, processes, assurance cases, field data, etc. Our extended socio-technical system captures the interde-

pendences of the socio and technical aspects. To reach that, when appropriate, within each layer (e.g., regulations, standards, etc.) we capture commonalities and variabilities. Inter-layer dependencies are also captured ensuring traceability and inter-layer variability management.

Ontology-based representation and SHACL constraints. In [3], we provided an ontology-based representation constituted of: 1) a feature model metamodel, formalized as an ontology with RDF (Resource Description Framework) [22], 2) a set of reusable constraints expressed with SHACL (Shapes Constraint Language) [23], which allow users to define "SHACL shapes", against which RDF graphs can be validated. Our SHACL constraints allow for characterising the nature of features, i.e., mandatory, optional, OR, XOR as well as for formulating cross-feature inclusion constraintsIt is worth to recall that SHACL constraints can be interpreted and processed by a SHACL engine, which checks and validates the SHACL constraints. In case there are facts in an RDF graph which violate any SHACL constraints, the SHACL engine reports a constraint violation, along with a description. This is then shown to the user. SHACL has its own vocabulary (e.g., a shape is specified with the construct sh:shape), which is formalised with RDF and used with standard ontology terms to define types, classes, subclasses, properties, lists and resources. It is also essential to recall that SPARQL (SPARQL Protocol and RDF Query Language) [21] can be used to formulate queries across diverse data sources (which can be stored natively as RDF graphs). Via this ontology-based representation, the connections among heterogeneous resources can be represented in a machine-understandable format.

3 Extended Ontology-based Representation

In this section, we expand our ontology-based representation with additional SHACL constraints for enabling 1) the auto-completion of feature trees (Subsection 3.1) and 2) the chaining of the implications from regulatory requirements to standard-related requirements down to the item (Subsection 3.2).

3.1 Rules for Auto-completion

In Listing1.1, we provide a reusable SHACL rule on the example of mandatory sub-features for enabling the auto-completion of feature trees. The idea behind is that whenever a feature has mandatory sub-features, and the feature has been instantiated (i.e. chosen), then the mandatory sub-features can be auto-generated as well, as they are mandatory and hence required. The following SPARQL-based SHACL rule realises that:

Listing 1.1. SHACL rule for the auto-completion applicable to mandatory features

```
co:SPARQLRule_AutoCreateMandatoryFeatures
  a sh:SPARQLRule ;
  sh:condition fm:Feature ;
  sh:construct """PREFIX sh: <http://www.w3.org/ns/shacl#>
PREFIX prov: <http://www.w3.org/ns/prov#>
CONSTRUCT {
```

```
    ?newFeatureNode a ?featureRangeClass.
    $this ?property ?newFeatureNode.
    ?newFeatureNode prov:generatedAtTime ?currentDateTime.
    ?newFeatureNode prov:wasGeneratedBy
         sa:MandatoryFeatureAutoGeneration.
    ?newFeatureNode prov:wasDerivedFrom $this.
}
WHERE {
    $this a ?featureSubclass.
    ?featureSubclass sh:property ?propertyShape.
    ?propertyShape sh:path ?property.
    ?propertyShape sh:class ?featureRangeClass.
    ?property rdfs:subPropertyOf fm:hasMandatoryFeature.
    FILTER NOT EXISTS {
       $this ?property ?value.
    }
    BIND (IRI(CONCAT("http://www.bosch.com/featuremodel#Feature_",
         STRUUID())) AS ?newFeatureNode)
    BIND(NOW() AS ?currentDateTime)
}""" ;
```

The presented SHACL rule is associated with the class `fm:Feature` and implicitly with all its subclasses, i.e., it applies to each feature instance. The `WHERE` clause as the rule body checks if the feature's class defines mandatory sub-features, by checking if any SHACL property shape for a sub-property of `fm:hasMandatoryFeature` exists. If so, it filters for those feature instances not having any such relationship defined yet. The `CONSTRUCT` clause as the rule head defines the implication of the rule, i.e. the new facts to be derived upon execution of the SHACL rule. In particular, it creates one new feature instance for each missing mandatory sub-feature and relates it to the given feature instance via the respective object property. This auto-completes the feature tree variant. Besides, for ensuring traceability, the new feature instance is associated with three provenance triples (lines 3-5 in the `CONSTRUCT` clause). Here we make use of the PROV Ontology (PROV-O), a W3C recommendation, which can represent, exchange, and integrate provenance information generated in different systems and under different contexts [19]. With provenance information we mean "information about entities, activities and people involved in producing a piece of data or a thing, which can be used to assess quality, reliability and trustworthiness" [20]. The provenance triples represent the date and time when the auto-completion happened (`prov:generatedAtTime`), the activity that triggered it (`prov:wasGeneratedBy`) and the feature instance it was derived from as mandatory sub-feature (`prov:wasDerivedFrom`). This provenance information helps identifying the features that were automatically generated by the rule, as well as their origin.

In Listing1.2, we provide another reusable SHACL rule on the example of required features for the auto-completion of feature trees. The rule is similar to the previous rule for mandatory sub-features, but applies to `fm:requiresFeature` relations. Requires relations are feature tree spanning, i.e. they point from a feature class of one feature tree to another one from another feature tree. Whenever a feature instance has been created, which requires another feature that is not yet present, the following rule can auto-generate a new instance of the required feature and hence satisfy the requires relationship. As with the previous rule,

the generated feature instance gets three provenance triples assigned (lines 2-4 in the `CONSTRUCT` clause), which help engineers tracing back when, how, and from what feature it got generated as well as which evidence shall be produced.

Listing 1.2. SHACL rule for the auto-completion applicable to requires relations

```
co:SPARQLRule_AutoCreateRequiredFeatures
  a sh:SPARQLRule ;
  sh:condition fm:Feature ;
  sh:construct """PREFIX sh: <http://www.w3.org/ns/shacl#>
PREFIX prov: <http://www.w3.org/ns/prov#>
CONSTRUCT {
    ?newFeatureNode a ?featureClass.
    ?newFeatureNode prov:generatedAtTime ?currentDateTime.
    ?newFeatureNode prov:wasGeneratedBy
          sa:RequiresFeatureAutoGeneration.
    ?newFeatureNode prov:wasDerivedFrom $this.
}
WHERE {
    $this a ?requiringFeatureClass.
    ?requiringFeatureClass fm:requiresFeature ?featureClass.
    FILTER NOT EXISTS {
    ?instanceFeatureClass a ?featureClass
    }
    BIND (IRI(CONCAT("http://www.bosch.com/featuremodel#Feature_",
          STRUUID())) AS ?newFeatureNode)
    BIND(NOW() AS ?currentDateTime)
}""" ;
```

3.2 Rule Chaining

Two examples of reusable SHACL rules for the auto-completion of feature trees were presented in the previous subsection. By executing the SHACL rules after each configuration step done by a user, all direct mandatory sub-features and all direct required features of all selected features will be auto-generated, along with provenance information. Executing the SHACL rules once, a one-step auto-completion is performed, i.e. the directly implied mandatory and required features are added only. Far more powerful is rule chaining, i.e., to trigger the auto-completion rules repeatedly in a chain, until no further new feature instances can be derived. This approach, also called forward chaining, derives all possible implications from a given set of facts over multiple steps, not just one. For example, from feature selected by a user, some mandatory sub-features might be automatically generated by executing the mandatory SHACL rule once. The generated feature instances may again require new mandatory or required features themselves. Triggering the auto-completion rules a second time will add the next feature instances, which in turn may again require additional features, that can be auto-generated by triggering the rules another time. This is repeated until no further features are generated. At this point, it is the user's turn to make the next choice, e.g. to select from optional, XOR or OR features.

Furthermore, it has to be noted here that the auto-generated feature instances are merely empty placeholders. The user needs to provide additional evidence for them, which can be stored via datatype properties. The feature classes of the feature ontology may each define their own datatype properties for providing different kinds of artefacts specific for the feature. This provides

guidance for the user what evidence and artefacts to provide, depending on the feature. By defining SHACL `minCount=1` constraints on the properties, a SHACL validation will show all the required but missing datatype properties of the feature instances. The user then is informed about it and can provide values for the properties, which are saved in the knowledge graph.

4 Compliance-aware System Change Management

In this section, we use our extended ontology-based representation to implement a system change management that is aware of regulatory compliance. To illustrate the effectiveness of the change management in supporting the user in analysing the impact of the change, we consider the case of the product line of WLs. We assume that safety engineers have provided the necessary evidence for showing that WLs with no external switch are in compliance with the safety standards (e.g., ISO 26262) and related regulation (e.g., UN21). We consider a system upgrade (increment). Specifically, the increment consists of the addition of a digital key, which enables the opening/closing of the vehicle but also the opening/closing of its windows. Based on what we recalled in the background, we know that the addition of a digital key implies a chain of requirements. Its addition requires the fulfilment of a set of specific legal requirements which imply the fulfilment of a set of requirements stated in the endorsed standards, which require the provision of specific evidence (e.g., damage scenarios, as part of TARA-related workproducts). Hence, due to the increment, previously developed evidence for compliance is no longer sufficient. The added feature makes the vehicle transit to a broader normative space, within the scope of other UNECE regulations (UN161, UN155) and ISO standards (ISO 21434). Precisely, from the background information, we can retrieve the following chain of implications: UNECER161.Annex9.5 requires UNECE155-Req5.1.1.b, which, in turn, requires an indirect endorsement of ISO 21434, which in turn, requires the execution of the ISO 21434 –TARA activities and the provision of the required deliverables. Hence, the evidence of the execution of ISO 21434 –TARA shall be in place, i.e., Threat scenarios, Damage scenarios, etc. Similar chains of implications could be retrieved. However, due to the space limitations, we focus on a single one. We also assume that the ISO 26262-related evidence remains relevant.

We show that our ontology-based system-change management is aware of regulatory compliance and capable to: 1) auto-generate mandatory and required, but missing features; 2) notify the user during the validation. For illustration, we follow the process in Figure 1. In what follows, first, we show the feature tree for the WLs, where the digital key is included. We also show the feature trees of the levels of the socio-technical system pertaining to the legal requirements and ISO standards (by exemplifying the feature tree for the cybersecurity standard). Then, we demonstrate the auto-generation. Figure 2 depicts the feature ontology model capturing the product line of WLs, described in Section 2.

Figure 3 depicts the feature ontology model capturing the legal requirements, based on the essential information, which was provided in the background.

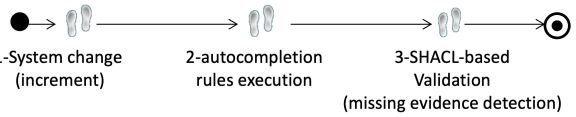

Fig. 1. Change impact analysis process represented in SPEM2.0 [14]

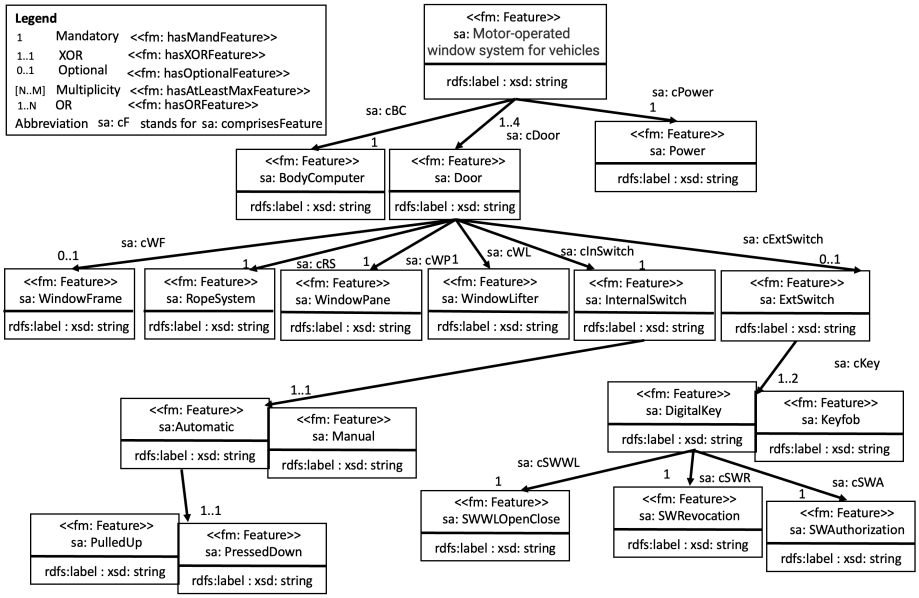

Fig. 2. A feature ontology capturing the WL-product line

Figure 4 shows the feature ontology model capturing the requirements of the cybersecurity standard, based on the essential information, which was provided in the background. We also modelled similar feature trees for the safety standards. However, for space reasons, we do not include them within this paper. Despite the existence of a shared glossary, due to the upcoming changes in automotive standards (e.g., expected publication of a new version of ISO 26262), we have modelled each standard as a separate feature tree.

Figure 5 shows the feature ontology model capturing the abstract interdependencies among the requirements of the different standards.

Auto-generation functionality: given a feature instance, several additional feature instances can be auto-generated with the SHACL rules introduced in Section 3. Figure 6 shows the auto-generated required feature instances with provenance information inferred from UNECE161ReqAnnex9.5 feature. More precisely, Figure 6 shows four automatically added features that were consecutively generated by the SHACL rule for required features: `UNECE155Req5.1.1.b`, `IndirectEndorsment`, `ISO21434Endorsement` and `TARAActivity`.

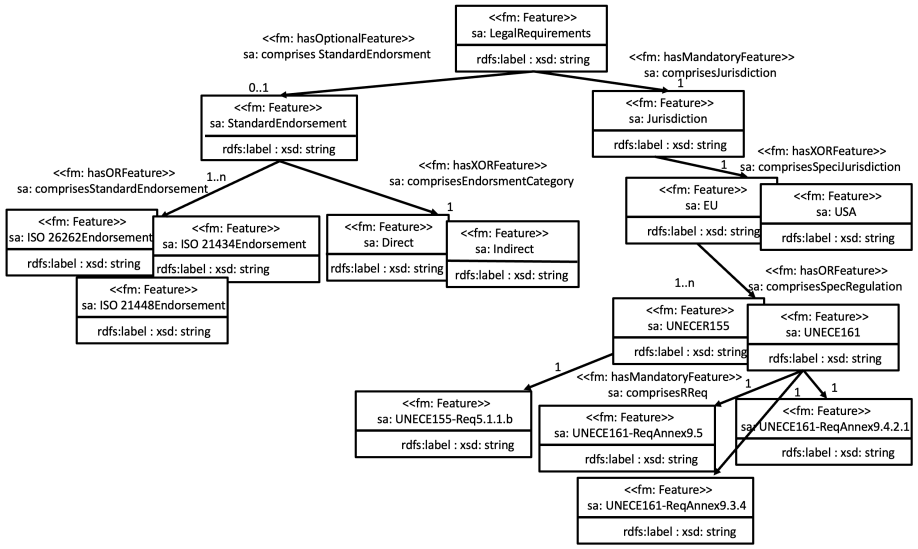

Fig. 3. A feature ontology capturing the legal requirements product line

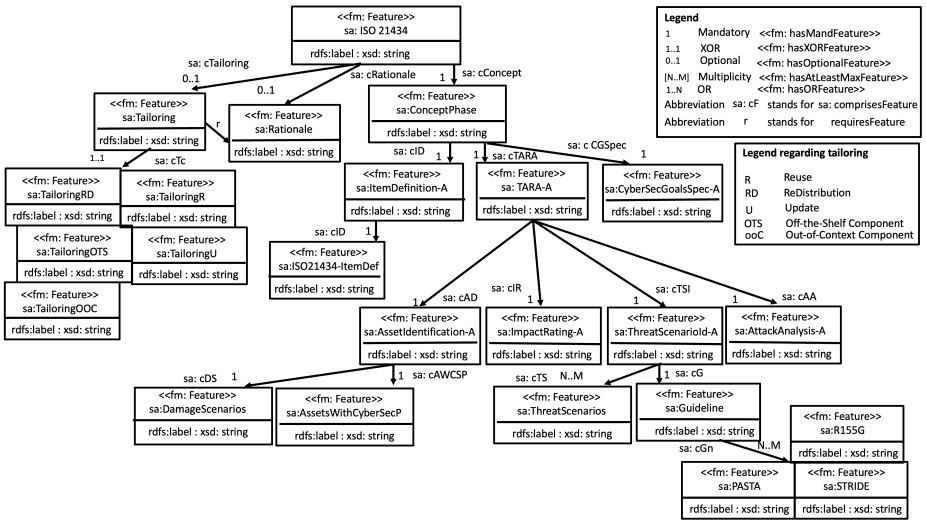

Fig. 4. A feature ontology capturing the ISO 21434 requirements product line

Furthermore, as shown in Figure 7, further feature instances can be auto-generated directly from `TARAActivity` with the SHACL rule for mandatory sub-features: `AttackAnalysisActivity`, `AssetIdentificationActivity`, `Threat ScenarioIdentificationActivity`, `ImpactRatingActivity`, and from there the additional features `AssetWithCyberSecProtection`, `DamageScenarios` and

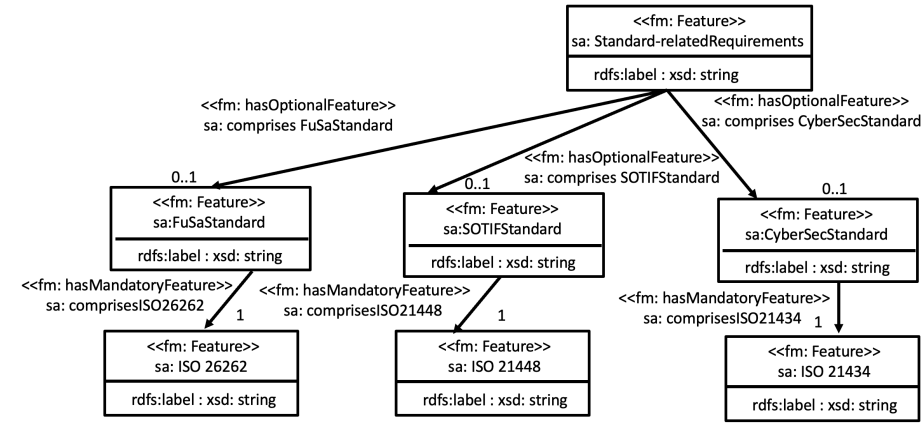

Fig. 5. A feature ontology capturing the standards-related requirements product line

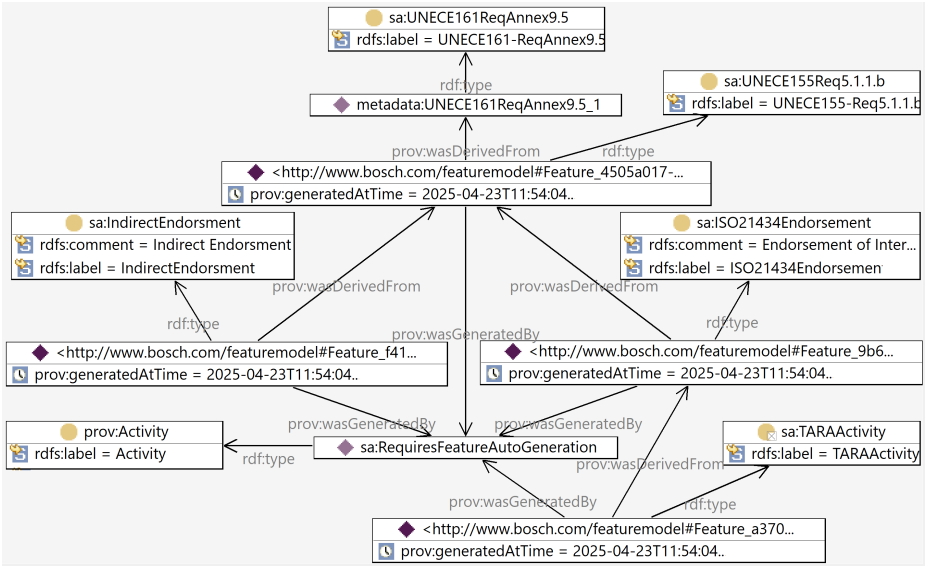

Fig. 6. Example of auto-generated required feature instances

`CybSecGuideline`. These autogenerated features constitute the basis for indentifying missing evidence (key-evidence omission fallacy) during the validation. Damage scenarios shall be provided, evidence is required.

Besides the generated feature instances, also provenance information is created, in particular the time stamps when they were generated (`prov:generatedAtTime`), the activity by which it was generated (`prov:wasGeneratedBy`), and the feature instance from which it was derived from (`prov:wasDerivedFrom`).

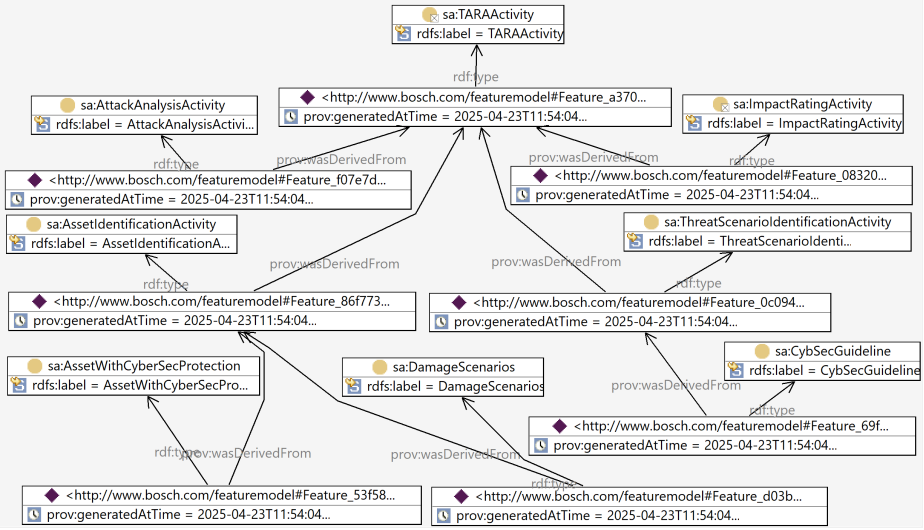

Fig. 7. Further example of auto-generated mandatory feature instances

5 Discussion and Synergy with the SPI Manifesto

In this section, we briefly discuss our findings based on the illustration of our ontology-based representation that enables system change management, aware of regulatory compliance. Compliance is much more than just the traceability of dependencies. However, since traceability plays a crucial role, we believe that our representation has the potential to be useful for tracing feature interdependencies related to any type of obligations traced down to the products and, where required, corresponding assurance cases. Assurance is much more than just traceability or detection of basic fallacies such as key-evidence omission. However, in this paper, we illustrate what can be automated toward more powerful change impact management in the context of assurance.

Regarding the synergy with the SPI Manifesto [12], which targets software, our extended ontology-based representation is not limited to software, but it embraces organisational change management in general and, in the context of this paper, it focuses on change management in relation to commonalities and variabilities among regulations, standards, technical products, etc. Our ontology-based representation is related to the SPI-principles: Create a learning organisation specifically on the interdependencies of the various socio-aspects with the technical aspects; Manage the organisational change in your improvement effort, specifically the variability management may support the three step model, i.e., “1) unfreeze, 2) move, 3) freeze”, which in our context can be interpreted as 1) explore the change via a new increment, 2) proceed with the increment, and 3) freeze once all evidence is in place.

6 Related work

In the literature, other works have focused on WLs from a functional (safety) specification's angle (e.g., [24].) or from a cybersecurity's angle [7]. However, as far as literature research revealed, no one has considered the socio-aspects that surround that system and no one has considered a product line of power-operated window systems. Regarding the exploitation of the Semantic Web stack, in previous work, Gallina et al. [5, 6] have proposed a layered ontology for capturing the Rasmussen socio-technical system. However, the variability management was not integrated. In [11], authors provide an RDF-based representation for capturing legal documents. Their representation, however, does not integrate a product line perspective. In addition, it only focuses on legal documents and it is not aimed at solving the compliance demonstration challenges of manufacturers.

7 Conclusion and Future Work

In this paper, we have extended our previously introduced ontology-based representation, which was shown to be effective for variability management in highly-regulated industries having market segments in different jurisdictions. The extension, provided in this paper, via the embedded regulatory compliance-awareness, represents an initial step towards the automatic management of system changes, while ensuring regulatory compliance. This step enables engineers to detect omitted evidence (Key-evidence omission fallacy). We have also illustrated the usage and effectiveness of the extended approach by considering a system upgrade in the context of a product line of motor-operated window systems. In future, we plan to further develop our approach by integrating it within the layered ontology presented in [6]. The integration shall allow us to reach the ontological infrastructure for a powerful evidence traceability and variability management solution, which can be extended/customised based on the application domain. Such infrastructure has the potential to cope with the increased complexity of the compliance demonstration.

References

1. Antinyan, V.: Revealing the complexity of automotive software. In: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. p. 1525–1528. ESEC/FSE 2020, Association for Computing Machinery, New York, NY, USA (2020)
2. Frederick, B.: No silver bullet essence and accidents of software engineering. Computer **20**(4), 10–19 (1987)
3. Gallina, B., Dibowski, H., Schweizer, M.: An ontology-based representation for shaping product evolution in regulated industries. In: Reuse and Software Quality: 21st International Conference on Software and Systems Reuse, ICSR 2024, Limassol, Cyprus, June 19–20, 2024, Proceedings. p. 92–102. Springer-Verlag, Berlin, Heidelberg (2024)

4. Gallina, B., Munk, P., Schweizer, M.: An extension of the rasmussen socio-technical system for continuous safety assurance. In: CARS@EDCC2024 Workshop - Critical Automotive applications: Robustness & Safety (2024)
5. Gallina, B., Olesen, T.Y., Parajdi, E., Aarup, M.: A knowledge management strategy for seamless compliance with the machinery regulation. In: Systems, Software and Services Process Improvement. pp. 220–234. Springer Nature Switzerland, Cham (2023)
6. Gallina, B., Steierhoffer, G.L., Young Olesen, T., Parajdi, E., Aarup, M.: Towards an ontology for process compliance with the (machinery) legislations. Journal of Software: Evolution and Process **37**(1), e2728 (2025)
7. Hoppe, T., Kiltz, S., Dittmann, J.: Security threats to automotive can networks—practical examples and selected short-term countermeasures. Reliability Engineering & System Safety **96**(1), 11–25 (2011), special Issue on SafeComp 2008
8. International Organization for Standardization: Iso 26262:2018 road vehicles – functional safety (2018)
9. International Organization for Standardization: ISO 21448:2022 Road Vehicles – Safety of the intended functionality (2022)
10. International Organization for Standardization / SAE International: ISO/SAE 21434:2021: Road vehicles — Cybersecurity engineering (2021)
11. Oliveira, Francisco de and Oliveira, Jose Maria Parente de: A rdf-based graph to representing and searching parts of legal documents. Journal of Artificial Intelligence Law **32**(3), 667–695 (Jul 2023)
12. Pries-Heje, J., Johansen, J. (eds.): MANIFESTO Software Process Improvement eurospi.net, Alcala, Spain (2010)
13. Slama, D., Lachenmaier, J.: Enabling applications for variable vehicle configurations with software-defined vehicle and service-oriented architecture. Procedia CIRP **126**, 289–294 (2024), 17th CIRP Conference on Intelligent Computation in Manufacturing Engineering (CIRP ICME '23)
14. SPEM 2.0: Software & Systems Process Engineering Meta-model (2008), http://www.omg.org/spec/SPEM/2.0/
15. ∞ COMPASS project. Team: ∞ COMPASS, Continuous Regulatory Compliance and Assurance of Socio-technical Systems #49, Software Center, https://www.software-center.se
16. UNECE: *Regulation 21 - Uniform Provisions Concerning the Approval of vehicles with regard to their interior fittings* (April 2003)
17. UNECE: *Addendum 154 – UN Regulation No. 155 - Uniform provisions concerning the approval of vehicles with regards to cyber security and cyber security management system* (January 2021)
18. UNECE: *Addendum 160 – UN Regulation No. 161 - Uniform provisions concerning the protection of motor vehicles against unauthorized use and the approval of the device against unauthorized use (by mean of a locking system)* (October 2022)
19. W3C: PROV-O: The PROV Ontology (2013)
20. W3C: PROV-Overview-An Overview of the PROV Family of Documents (2013)
21. W3C: SPARQL 1.1 Query Language (2013)
22. W3C: RDF 1.2 Concepts and Abstract Syntax (2014)
23. W3C: Shapes Constraint Language (SHACL), W3C Recommendation (2017)
24. Xu, Y., Li, Y., Li, C.: Electric window regulator based on intelligent control. Journal of Artificial Intelligence and Technology **1**(4), 198–206 (Sep 2021)