Towards an ISO 26262-compliant OSLC-based Tool Chain Enabling Continuous Self-assessment

Barbara Gallina
MRTC, Mälardalen University,
Västerås, Sweden
Email: barbara.gallina@mdh.se

Kathyayani Padira and Mattias Nyberg
Scania AB
Södertälje, Sweden
Email: {kathyayani.padira,mattias.nyberg}@scania.com

***Abstract*—Manufacturers of road vehicles have to comply with the functional standard ISO 26262. This standard requires the creation of a safety case, which compiles all the work products of the life-cycle in a traceable manner. The creation of a safety case is extremely time-consuming. Its creation is needed for the purpose of self-assessment in order to manage the liability risk. To speed up such creation, semi-automatic generation represents an interesting solution. OSLC (Open Services for Lifecycle Collaboration) is a recently introduced standard aimed at enabling life cycles tools interoperability via production/consumption of resources. OSLC defines a set of domains, each of which focuses on a single and generic phase of the life-cycle. To create an ISO 26262-compliant tool chain, such domains need to be extended and or replaced. In this paper, we present the first step towards the extension of such domains. First we establish the guidelines to define an ISO 26262-compliant OSLC-based knowledge domain, aimed at enabling the production and consumption of resources related to verification work products. Then, we propose our domain and we instantiate it to represent the verification-related resources of a real system. Finally, we discuss how this domain has been validated and how it can be used for continuous self-assessment.**

Keywords*—*Safety cases, ISO 26262, OSLC, Interoperability, Tool Chain, Quality Management Domain, Knowledge Representation, Continuous Self-assessment.

I. Introduction

ISO 26262 [9] is a relatively new standard for functional safety in the automotive domain. This standard defines a safety life-cycle for the development of safety-critical systems. This life-cycle is expected to be shaped according to a V-model. To show that an item (system under examination) is acceptably safe to operate in the expected context, a safety case must be created. According to ISO 26262- Part 1, Definition 1.106, a safety case is an argument that claims about completeness and satisfaction of the safety requirements for an item. Claims are supported by evidence compiled from work products of the safety activities performed during the item development.

As already recalled in our previous work [7], which was focused on documentation management-related challenges, ISO 26262 also imposes a set of requirements for managing the documentation. Managing the documentation represents a relevant activity especially in safety-critical systems engineering. The documentation process is usually tightly coupled with the development life-cycle. Life-cycle's work products represent immediate as well as direct evidence to be used during the safety assessment process to support the claims about system's safety. More specifically, the left-hand side work products of the V-model (e.g., requirement specification) represent immediate evidence; while the right-hand side work products of the V-model (e.g., verification results) represent direct evidence. Improper documentation/evidence management may indirectly result in certification risk.In the context of ISO 26262, for instance, the goal of the documentation process is to make documentation available: 1) during each phase of the entire safety lifecycle for the effective completion of the phases and verification activities; 2) for the management of functional safety, and 3) as an input to the functional safety assessment.

Initially introduced for road vehicles up to 3,5-ton gross mass, ISO 26262 is now under revision to be proposed for all road vehicles, including heavy trucks. A new version of the standard is expected to be issued by 2018. As already highlighted in our previous work [7], manufacturers of heavy trucks are keeping a constant eye on the ongoing revision of the standard and they are strategically planning its timely adoption by 2018 [3], [6]. As envisioned in our previous work [7], to manage the documentation in compliance with the standards and to be able to generate semi-automatically the required safety case for the purpose of self-assessment, an OSLC-based interoperable tool chain could represent the winning solution. In this paper, we perform the first step towards the concretisation of such tool chain. More specifically, in this paper, first, we provide a set of general methodological guidelines to be able to extract from normative documents the resources, needed to define new domains as extensions of OSLC pre-existing domains. Then, we focus our attention on a portion of direct evidence stemming from the application of clauses 9-11 of Part 6 of ISO 26262. Based on our methodological guidelines, we provide a knowledge domain for such evidence. Our proposed domain is an ISO 26262-compliant extension of the OSLC domain for quality management. Then, we instantiate our domain to represent the knowledge (more specifically, verification-related evidence) associated to a Scania ECU (Electronic Control Unit). Consequently, we conduct an empirical validation. Finally, we explain how our domain can be used for continuous self-assessment via the creation continuous creation of safety cases. The rest of the paper is organised as follows. In Section II, we provide essential background information. In Section III we present the proposed ISO 26262-compliant and OSLC-based domain aimed at supporting interoperability of quality management information. In Section IV, we instantiate our domain for representing information related to the testing of a Scania ECU. In Section V, we conduct the validation of our proposed domain. In Section VI, we discuss how our domain can be used for continuous self-assessment. In Section VII, we discuss related work. Finally, in Section VIII we present some concluding remarks and future work.

II. Background

In this section, we present the background information on which we base our work. In particular, in Section II-A, we provide a brief overview of ISO 26262 and detailed information about Part 6-product development at the software level. In Section II-B we provide an overview of OSLC and its underlying set of specifications, necessary to define ontological domains. In Section II-C we provide essential information related to the ECU contained in safety-critical system, used to show the domain instantiation.

A. ISO 26262

ISO 26262 regulates all phases of the entire lifecycle of the product (item), starting from the management and requirements specification phases up to the production release. The standard recommends the usage of a V-model at item level as well as at element (software and hardware) level. ISO 26262 consists of 9 normative parts, each of which structured into clauses. All the clauses state the objectives, inputs for the clause, recommendations and requirements to be fulfilled and finally the work products that are to be generated. Notes are also included. Notes are not normative and are expected to help the applicant in understanding and interpreting the requirements. Additionally, obligations on the corresponding methods are also imposed based on the assigned ASIL. In this paper, we limit our attention to a subset of clauses (9-11) of Part 6 that are related to the right side of the software V-model, as depicted in Fig. 1 adapted from [9]. In this paper, the clauses 6-8 in part 6 of the standard are out of scope (i.e the left hand side of the software V-model). However, they are depicted to make visible how the two arms of the V-model are related.

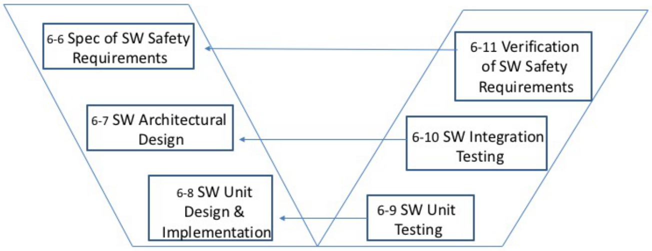
6-6 Spec of SW Safety Requirements
6-7 SW Architectural Design
6-8 SW Unit Design & Implementation
6-11 Verification of SW Safety Requirements
6-10 SW Integration Testing
6-9 SW Unit Testing

Fig. 1: Zoom on the ISO 26262 V-model related to Part 6.

These clauses are:

9 **Software unit testing** - The main objective of this clause is to verify that the implemented software units are as per the software unit design specification and do not contain any undesired behaviour.

10 **Software integration and testing** -The objectives of this clause are to integrate the software units and generate the embedded software and to verify that the embedded software is as per the software architectural design and does not contain undesired behaviour.

11 **Verification of Software safety requirements** -The main objective of this clause is to verify that the developed embedded software satisfies all the software safety requirements defined in Part 6, clause 6.

All the above-listed clauses require the generation of three work products: software verification report (SVR), software verification specification (SVS) and software verification plan (SVP). For sake of completeness is to should be mentioned that the standard recommends best practices for achieving functional safety by recommending the generation of hundreds of work products, however it also defines tailoring rules which can be applied to omit the generation of some work products in case of a justified and well defined rationale.

B. OSLC

OSLC [13] is an industrial standard that targets tools used during a products life cycle and enables their integration and interoperability. Tools for requirements engineering, design, implementation, verification, etc. are expected to interoperate in a traceable manner i.e. traceability between the respective work products can be easily retrieved and shown. To enable interoperability, different specifications, called domains, need to be provided. More precisely, an OSLC Domain is one ALM (Application Lifecycle Management) or one Product Lifecycle Management (PLM) topic area such as Quality Management (QM), Architecture Management (AM), Requirements Management (RM). Each OSLC Domain has its own OSLC specification that complies with this core specification. OSLC builds on top of Linked Data [12], Resource Description Framework (RDF) [15], RDF Schema [16], and HTTP protocol. Each work product is described as an HTTP resource, identified via a Uniform Resource Identifier (URI). Work products are manipulated via HTTP methods (i.e., GET, POST, etc.). To interoperate via a work product, a tool that acts as a provider has to associate an URI to the work product and post it; a tool acting as consumer can get the work product via the URI. RDF provides a standard representation for data as directed graphs to facilitate the linking of the resources to be described. RDF Schema provides a data-modelling vocabulary for RDF data. RDF Schema is an extension of the basic RDF vocabulary. RDF Schema is complemented by several companion documents which describe the basic concepts and abstract syntax of RDF as well as its formal semantics. The core structure of linked data, presented in form of RDF-graphs, consists of triples. These triples consist of subject, object and predicate (refer to Fig. 2). RDF allow only binary relationship to be represented.

Subject
Predicate
Object

Fig. 2: Linked Data Triple.

C. CMS (Chassis Management System)1

In this section, we provide essential information concerning CMS1 and its context. CMS1 is an ECU (Electronic Control Unit), which along with other ECUs is used for realising the Fuel Level Estimation and Display System (FLEDS) functionality within Scania products.

The main functionality of a FLEDS consists of the estimation of the fuel level in the vehicle tank and presentation of this level on the display located in the dashboard. Additionally, the driver must be warned if the fuel level is below a predefined level. These functionalities are deployed onto different ECUs. These two main functionalities are assigned to logical software elements called Allocation Element 201 (AE201) and Allocation Element 202 (AE202) respectively at Scania. CMS1 is used for realizing both AE201 and AE202 and it is responsible for calculating the total fuel level. Within CMS1 there is a software module, called CMS1: Fuel. In Section IV, we limit our attention to this module and more specifically to the work products generated during the software unit testing. CMS1 used in FLEDS is an example of a typical case for software testing performed for large number of ECUs developed at Scania. Thus, CMS1 can be considered a representative case.

III. ISO 26262-COMPLIANT QM EXTENSION

This section builds on top of the master thesis work that was conducted at Scania [14]. In this section, first, we present a set of general methodological guidelines to be able to define new domains targeting ISO 26262 parts. Then, we present our proposal for an OSLC-based domain targeting ISO 26262, Part 6, Clauses 9-11. To introduce a new domain, two options are at disposal either a new domain is introduced from scratch or a new domain is obtained via extension of a pre-existing one. When a domain already offers interesting resources, the second option is to be recommended. In both cases, before applying OSLC best practices, guidelines are needed to proceed. We have elaborated a set of guidelines to be applied systematically to each clause in order to identify and model the needed resources. These guidelines are:

- identify the work product types that are required and are expected to be compiled during the creation of a safety case. As mentioned in Section II-A, each clause in the ISO 26262 standard is structured in the same way and it contains: objectives, general, inputs for the clause, requirements and recommendation, and work products. Thus, the identification of the work product types is straightforward since the list of work products is clearly defined. For each listed work product, a corresponding work product type is defined and modelled as a meta-class.
- identify the text that describes relevant information for characterising the work products. Once the text is identified, meta-attributes and/or other types are added in order to fully characterise the work product types. that are used to describe these three work products types are defined by analysing the requirements and recommendations sub clause.
- identify the text that describes associations that inter-relate the work products. Once the text is identified, meta-associations can be added to inter-relate the work product types.

As mentioned in Section II-B, QM is an OSLC domain for Quality Management. QM defines QM resources such as Test Plan, Test Case, Test Script, Test Execution Record, and Test Result. Since some of these resources are of interest, we opt for extending the pre-existing QM. To do that, we first apply the above-listed guidelines to Part 6, Clauses 9-11. We identify the work products that are required by each clause. These work products are: SVS, SVP and SVR. Thus, three corresponding work products types are defined and modelled as meta-classes. Then, we identify all the information related to SVS, SVP and SVR. This information is used to define meta-attributes as well as additional types, modelled as additional meta-classes. For instance, an SVP must be characterised by the information related to the pass criteria, the level, the regression strategy. Finally, meta-association are used to relate all the meta-classes. For instance, SVR is generated based on SVS, which in turn is based on SVP.

As previously mentioned, these methodological guidelines should be applied to each clause. By doing so, and by exploiting the information related to the expected input of each clause, it is possible to establish the meta-association that relate one domain (e.g., QM) with other domains (e.g., AM and RM). Thus, it is possible domain after domain to reproduce an OSLC-based representation of software V-model which was recalled in Fig. 1.

As a result, a set of inter-related meta-classes representing the targeted ISO 26262-compliant QM-resources is obtained. For sake of readability and communication, we suggest to first depict the domain, as done in Fig. 3, as class diagram in compliance with a UML profile for OSLC as proposed in [21]. This profile permits users to stereotype a class by indicating that it is an OSLC resource. It also permits users to specify typical attributes (e.g., title, identifier) in compliance with the Dublin Core (DC) Vocabulary [4]. Note that in Fig. 3 not all the DC terms have been represented. The depicted meta-model is given in a human-readable format, which can be easily discussed with a set of experts to get their approval. Then, we proceed with the manual translation of the UML-based representation into an OSLC-based domain, given as RDF-Schema. Once the schema is created, an instance can be instantiated, populated, and represented as an RDF-graph. It is worth to note that this translation could be automated as explored in [19]. To properly shape RDF-graphs, the results of the ongoing work [5] are expected to be integrated.

IV. INSTANTIATING OUR ISO 26262-COMPLIANT QM

In this section, we instantiate our ISO 26262-compliant QM for the Scania software module CMS1:Fuel. To do that, we collect via interviews the information needed for identifying and defining data for populating the instance of the meta model with respect to the CMS1. The interviewees are asked questions like “What is a plan for verifying ECU softwares?”, “What are the smallest software parts you test for?” etc. Questions are made to make interviewees think aloud their thoughts as suggested in [11]. Thus, different observations are made without the researcher getting involved in the activity being observed. Further, for mapping the terms and concepts between ISO 26262 and Scania, we use the findings of a previous master thesis [18], which was aimed at performing the gap analysis between the Scania process and the ISO 26262 life-cycle. This usage of this mapping is needed for formulating interview questions using terms which are familiar to the interviewees in order to avoid misunderstandings. The used mapping is given in Table I.

To instantiate our ISO 26262-compliant QM for the Scania software module CMS1:Fuel, we also consider other docu-

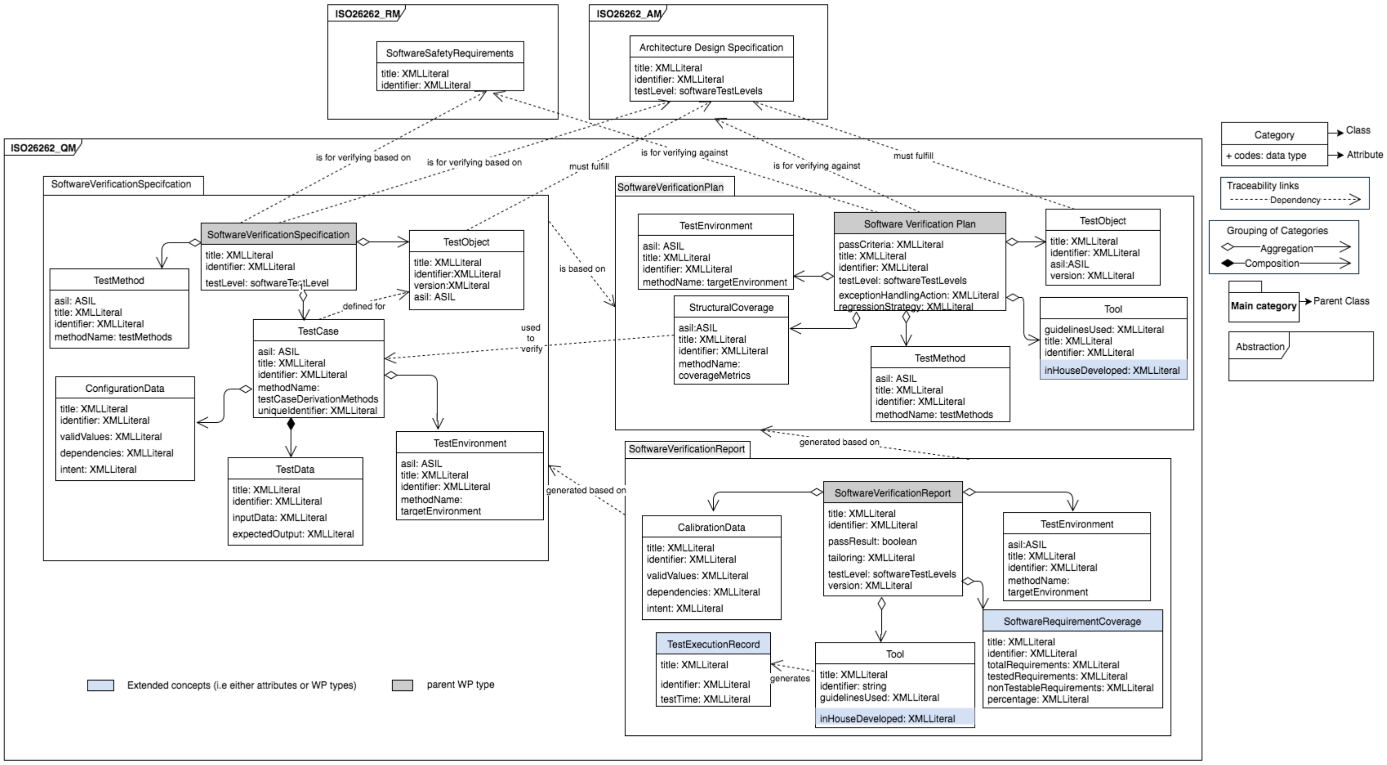

Fig. 3: ISO 26262-compliant QM depicted as an UML Class Diagram.

TABLE I: Mapping of ISO 26262 and Scania Terms.

Scania Term	ISO 26262
CMS1	Item
Element	Any part of the CMS1 (i.e software unit, component etc)
Module Testing	Software Unit Testing
Module Integration Testing	Software Integration and Testing
Allocation Element Requirement (AER)	Technical Safety Requirement
Assumption	
some AERs	Software Safety Requirements

ments, i.e., test cases, configuration data, software verification reports, design specifications, requirements specifications involved in software testing of CMS1. Complemented by interviews, these documents help us in identifying and defining the values for the attributes of the work products. Access to these documents has been been obtained via the concerned interviewee or via the official document management system at Scania.

The instance of the portion of the domain representing the Software Verification Plan for the software module, called CMS1: Fuel, is depicted in Fig.4. This plan is related to the software unit testing and it states that as exception handling strategy, the tool Jira should be used to keep track of eventual bugs. It also defines the criteria for pass. The purpose of Fig.4 is not be fully representative but simply to show a human-readable instance.

The corresponding machine-readable RDF-graph is depicted in Fig.5.

V. Validation

To validate our domain, we perform an empirical validation. First, we define a set of criteria (traceability, confirmability and abstraction). Then, we prepare a questionnaire and we use it to interview a representative sample of the population that might constitute the users of our proposed domain. We evaluate the answers against the criteria and we draw our conclusions. The respondents range from individuals having basic knowledge of ISO 26262 to experts. The varied experience and expertise level of the subjects helps to adjudge the effectiveness of the conceptual meta model encompassing for a comprehensive sample population. From the validation results, it can be concluded that the proposed domain has been effective to a great extent in achieving compliance to ISO 26262. The respondents also seemed to be positive with respect to our pioneering approach and currently there is an interest to build on top of it to bring it a step further: from a proof of concept-related work to a solution to be used in production-settings. Respondents have perceived this work as necessary in a mid-long-term future. However, they also realised the extra effort that this OSLC-based infrastructure would require. Additional details concerning the validation process and results can be found in [14].

VI. Continuous Self-assessment

The proposed ISO 26262-compliant QM constitutes a tile of the entire mosaic, expected to represent an ISO 26262-compliant OSLC-based Tool Chain. Via our proposed QM, once appropriate OSLC-adaptor are implemented, testing tools at Scania will be able to expose the needed resources to be consumed by other tools within the tool chain. One of

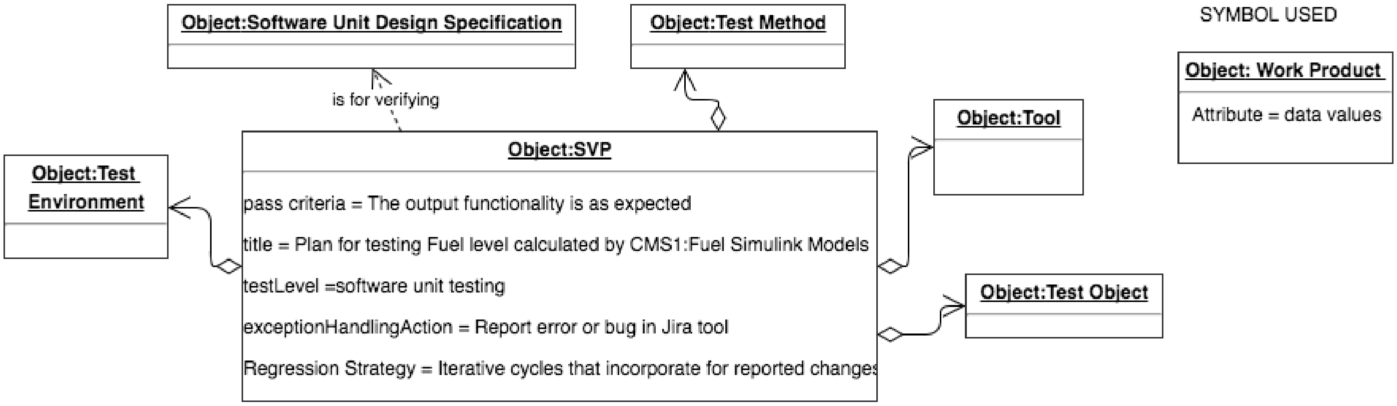
Object:Software Unit Design Specification
Object:Test Method
SYMBOL USED
Object: Work Product
Attribute = data values
is for verifying
Object:Tool
Object:SVP
Object:Test Environment
pass criteria = The output functionality is as expected
title = Plan for testing Fuel level calculated by CMS1:Fuel Simulink Models
testLevel =software unit testing
exceptionHandlingAction = Report error or bug in Jira tool
Object:Test Object
Regression Strategy = Iterative cycles that incorporate for reported changes

Fig. 4: Instance representing the SVP.

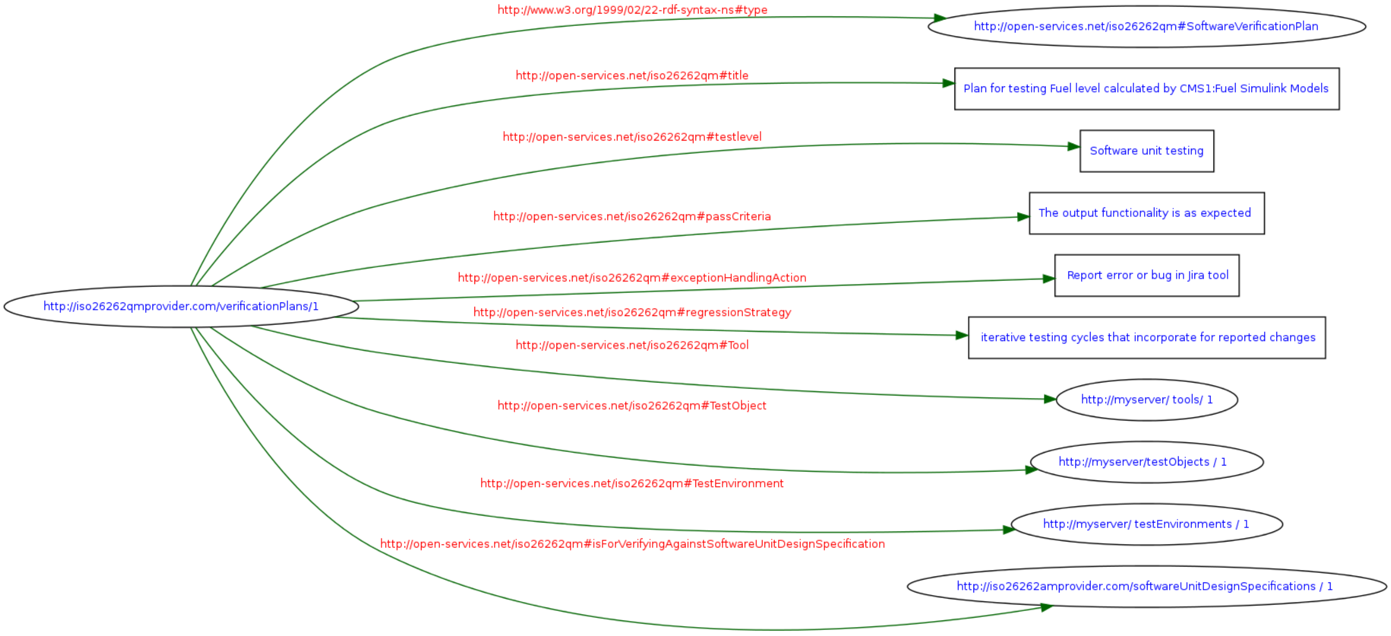
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://open-services.net/iso26262qm#SoftwareVerificationPlan
http://open-services.net/iso26262qm#title
Plan for testing Fuel level calculated by CMS1:Fuel Simulink Models
http://open-services.net/iso26262qm#testlevel
Software unit testing
http://open-services.net/iso26262qm#passCriteria
The output functionality is as expected
http://open-services.net/iso26262qm#exceptionHandlingAction
Report error or bug in Jira tool
http://iso26262qmprovider.com/verificationPlans/1
http://open-services.net/iso26262qm#regressionStrategy
iterative testing cycles that incorporate for reported changes
http://open-services.net/iso26262qm#Tool
http://myserver/ tools/ 1
http://open-services.net/iso26262qm#TestObject
http://myserver/testObjects / 1
http://open-services.net/iso26262qm#TestEnvironment
http://myserver/ testEnvironments / 1
http://open-services.net/iso26262qm#isForVerifyingAgainstSoftwareUnitDesignSpecification
http://iso26262amprovider.com/softwareUnitDesignSpecifications / 1

Fig. 5: Corresponding RDF graph for the SVP Work Product.

this tools is the safety case generator, which is expected to consume the exposed information by querying the various RDF-graphs. Once the information is retrieved, a safety case can be generated as a compilation of work products. As it was modelled, a software verification report is constrained by a software verification specification which in turn has to stick to the software verification plan. This represents an example of a structural traceable compilation needed for self-assessment with respect to ISO 26262 compliance. The limited world that we have considered can and should be expanded to embrace the other life-cycle phases and thus build the entire story that permits a safety engineer to argue about system safety via demonstration of completeness of requirements with respect to the identified hazards; decomposition, implementation and verification of requirements. Given the OSLC-based infrastructure, the generation of the safety case can be done continuously permitting to monitor its progress: from a preliminary and skeleton-oriented version to a complete and operational one.

VII. Related Work

The exploration and exploitation of OSLC has been part of many research agendas. Various EU projects (e.g., CESAR, MBAT, iFEST, and CRYSTAL) have extensively used and contributed to further develop OSLC in order to enable tools interoperability. In the context of such projects, to the best of our knowledge, however, no work has been proposed to offer methodological guidelines for the extension of OSLC-domains in compliance with normative standards. More specifically, no work has been proposed to exploit OSLC for enabling the semi-automatic generation of safety cases for the purpose of self-assessment. More recently, in the medical domain [17], a Process Assessment Model (PAM) based on ISO 15504 is proposed. The traceability requirements between the risk management and change management processes within the software development process are identified. After having identified the traceability requirements, authors envision the possibility to automate the generation of a safety case via the exploitation of the OSLC specifications. The vision is discussed but no concrete step is carried out. In [10], the authors focus their attention on not only enabling interoperability via OSLC but also on representing knowledge in a more accurate way. To do that, they propose an OSLC Knowledge Management specification and a mapping between RDF and RelationSHiP, which is an alternative of RDF that allows N-ary relationships to be represented. Accuracy is crucial also in our case. However, the main focus of our unit of work is not on how to represent but on what to represent.

VIII. Conclusion and Future Work

In this paper, we have presented the first step towards an ISO 26262-compliant OSLC-based tool chain enabling continuous self-assessment, via the continuous semi-automatic creation of safety cases. Our first step consisted of the proposal of an OSLC-based domain targeting ISO 26262, Part 6, Clauses 9-11, together with a set of general methodological guidelines to be able to define new domains targeting other parts or more generally other normative documents. Our proposed domain supports the compilation of traceable work product types towards the semi-automatic creation of a safety case. To empirically investigate the effectiveness of our domain, we have instantiated it for a truck ECU system at Scania. An RDF-graph was created. Finally, a validation was conducted.

In a short-term future, first of all, we aim at applying our methodological guidelines for the extension of OSLC domains aimed at targeting other parts of ISO 26262. In parallel, in addition to the validation presented in this paper, we intend to conduct a more in depth validation. More specifically, based on a pre-existing validation approach [1], we intend to compare our proposed domain with empirically gathered concept maps of individuals of different knowledge level, including experts. To do this a new iteration of interviews will be needed.

In medium term future we aim at creating the OSLC-compliant infrastructure necessary to enable tool interoperability. Finally, we plan to achieve a proof of concept concerning continuous self-assessment by generating argument fragments based on the information extracted from the ISO 26262-compliant RDF-graphs.

In a long term future, once the proof of concepts succeeds, we aim at considering additional but compatible representation means aimed at improving the accuracy of the represented knowledge.

Acknowledgment: This work has been partially financially supported by the Swedish Foundation for Strategic Research via the SSF Gen&ReuseSafetyCases project [8] project. This work has also benefited from the discussions held in the context of the ECSEL Joint Undertaking project AMASS (No 692474) [2] as well as the ESPRESSO project [20]. We also thank the Scania engineers, who have patiently answered to our multiple questions.

References

[1] D. Albert and C. M. Steiner. Representing domain knowledge by concept maps: How to validate them? In T. Okamoto, D. Albert, T. Honda, and F. W. Hesse, editors, *The 2nd Joint Workshop of Cognition and Learning through Media–Communication for Advanced e–Learning*, pages 169–174, 2005.

[2] AMASS (Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems). http://www.amass-ecsel.eu.

[3] R. Dardar, B. Gallina, A. Johnsen, K. Lundqvist, and M. Nyberg. Industrial experiences of building a safety case in compliance with iso 26262. In *IEEE 23rd International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 349–354, 2012.

[4] Dublin Core Metadata Initiative. http://dublincore.org/documents/dcmi-terms/.

[5] B. Gallina, J. P. Castellanos Ardila, and M. Nyberg. Towards Shaping ISO 26262-compliant Resources for OSLC-based Safety Case Creation. In *Critical Automotive applications: Robustness & Safety (CARS), Gteborg, Sweden, HAL*, September 2016 (in press).

[6] B. Gallina, A. Gallucci, K. Lundqvist, and M. Nyberg. VROOM & cC: a Method to Build Safety Cases for ISO 26262-compliant Product Lines. In *2nd Workshop on Next Generation of System Assurance Approaches for Safety-Critical Systems*. Hyper Articles en Ligne (HAL), September 2013.

[7] B. Gallina and M. Nyberg. Reconciling the ISO 26262-compliant and the Agile Documentation Management in the Swedish Context. In *Critical Automotive applications: Robustness & Safety (CARS), Matthieu Roy, Paris, France, HAL*, September 2015.

[8] Gen&ReuseSafetyCases-SSF-SM140013. http://www.es.mdh.se/projects/393-genreusesafetycases.

[9] International Organization for Standardization (ISO). ISO 26262: Road vehicles — Functional safety, 2011.

[10] J. L. Jose Mara Alvarez-Rodrguez, Manuela Alejandres and J. Fuentes. OSLC-KM: A knowledge management specification for OSLC-based resources. *25th Annual INCOSE International Symposium (IS) Seattle*, 25(1):16–34, 2015.

[11] T. C. Lethbridge, S. E. Sim, and J. Singer. Studying software engineers: Data collection techniques for software field studies. *Empirical software engineering*, 10(3):311–341, 2005.

[12] Linked Data. http://www.w3.org/designissues/linkeddata.html.

[13] Open Services for Lifecycle Collaboration. http://open-services.net/.

[14] K. Padira. Investigation of Resources Types for OSLC domains Targeting ISO 26262: Focus on Traceable safety evidence for the Right side of the ISO 26262 Software V-model. Master's thesis, Blekinge Tekniska Hgskola, Karlskrona, Sweden, to appear in 2016.

[15] RDF Primer. http://www.w3.org/tr/rdf-primer/.

[16] RDF Schema 1.1, W3C Recommendation 25 February 2014. http://www.w3.org/tr/rdf-schema/.

[17] G. Regan, M. Biro, D. Flood, and F. McCaffery. Assessing traceabilitypractical experiences and lessons learned. *Journal of Software: Evolution and Process*, 27(8):591–601, 2015.

[18] J. Reineck and J. Westman. Gap analysis on software development between ISO 26262 and scania. Master's thesis, Uppsala University, Uppsala, Sweden, 2011.

[19] Q. Tong, F. Zhang, and J. Cheng. Construction of RDF(S) from UML Class Diagrams. *Journal of Computing and Information Technology (CIT), doi:10.2498 /cit.1002459*, 22(4):237–250, 2014.

[20] VINNOVA, 2011-04446-ESPRESSO. http://www.vinnova.se/sv/resultat/projekt/effekta/espresso/.

[21] W. Zhang. Class Modeling of OSLC Resources. Technical report, University of Oslo, Norway, Report no-428, ISBN 82-7368-392-3, 2013.