Safety Case Development with SBVR-Based Controlled Language

Yaping Luo[1], Mark van den Brand[1(✉)], and Alexandre Kiburse[2]

[1] Eindhoven University of Technology,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{y.luo2,m.g.j.v.d.brand}@tue.nl
[2] Klee Group, La Boursidière, 92350 Le Plessis-Robinson, France
alexandre.kiburse@kleegroup.com

Abstract. Safety case development is highly recommended by some safety standards to justify the safety of a system. The Goal Structuring Notation (GSN) is a popular approach to construct a safety case. However, the content of the safety case elements, such as safety claims, is in natural language. Therefore, a common understanding of the meaning of a safety claim may be difficult to reach. Consequently, the confidence of a safety claim can be misplaced. In this paper, we propose to use an SBVR-based controlled language to support safety case development. By using the controlled language, the ambiguities caused by natural language can be mitigated. Furthermore, an SBVR editor for building a vocabulary and a GSN editor with vocabulary support are developed. Finally, a case study has been carried out to show the benefits of using the controlled language for safety case construction.

Keywords: Safety case · SBVR · Controlled language · Conceptual model

1 Introduction

A safety case is a well-structured argument for justifying that systems are safe. A safety case is defined as: "a documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment" [6]. It is used to show that a system, service or organization will operate as intended for a defined application in a defined environment. In some international safety standards, explicit safety cases are required for safety-critical systems. For example, ISO 26262 [13] in the automotive domain and DO 178C [4] in the avionic domain, stimulate the use of safety cases to demonstrate the product safety [24]. Besides, MOD Def Stan 00-55 [21] for safety-critical software in defense equipment requires producing safety cases with explicit safety requirements. Due to these characteristics, a safety case must be represented in a correct and understandable structure. Thus, the safety case should be carefully developed and defined.

P. Desfray et al. (Eds.): MODELSWARD 2015, CCIS 580, pp. 3–17, 2015.
DOI: 10.1007/978-3-319-27869-8_1

There are mainly two ways to document a safety case: textual and graphical. For textual safety case documents, the logic and structure of a safety case are implicit [2]. This could bring inconsistencies and confusion, and thus the understandability of the safety case can be harmed. For graphical safety case documents, the Goal Structuring Notation (GSN) has been proposed for representing the argument structure [14]. It provides a number of graphical symbols to assist the development of safety cases. By using GSN, the structure of a safety case is explicit and clear. However, as more and more users (argument readers and writers, such as safety engineers, or safety assessors) are involved in safety case development, shared understanding of the meaning of safety case elements are important. If the shared understanding is missing, the confidence of a safety case can be misplaced.

To address this, some research has been done on the understandability of safety arguments. In [12], assured safety arguments are proposed as a clear argument structure to demonstrate how to create clear safety arguments. Besides, in [11], a precise definition of the context in GSN arguments is proposed to achieve a better understanding. However, the content of a safety case element is still documented in natural language. The ambiguities caused by natural language are still unsolved.

In our previous research, we proposed a methodology to use an Semantics of Business Vocabulary and Rules (SBVR) based controlled language [23] to support the development of clear safety arguments [18]. By using a controlled language, all the concepts (noun concepts and verb concepts) in a safety case are well-defined in an SBVR vocabulary. Argument readers can check the definitions or examples of those concepts to get a shared understanding of them. In this way, the understandability of safety arguments can be improved. However, the process of our previous methodology can only be done manually. To address this issue, in this paper we propose a model transformation to generate SBVR vocabularies from EMF conceptual models. This reduces the manual work involved in vocabulary development. Moreover, we built two editors to facilitate the safety case construction. An SBVR editor is implemented for modifying or creating a vocabulary, and a GSN editor is developed to enable argument writers to edit GSN elements in the controlled language. Finally, we carried out a case study to show the benefits of using the controlled language for construction safety cases.

The remaining paper is organized as follows: Sect. 2 introduces GSN and SBVR. Section 3 presents the three main phases in our vocabulary-based methodology for safety case development. Section 4 shows a case study on two existing safety cases. Section 5 discusses the related work of this paper. Finally, Sect. 6 summarizes our conclusions and future work.

2 Background

In this section the basic information used in the remainder of this paper is discussed. We give a brief description of Goal Structuring Notation (Sect. 2.1) and Semantics of Business Vocabulary and Rules (Sect. 2.2).

2.1 Goal Structuring Notation

Goal Structuring Notation (GSN) is a graphical notation which is widely recommended for presenting safety cases [15]. It provides a clear and well-structured argument in terms of basic graphical elements, such as goals, solutions and strategies. As mentioned before, safety standards such as ISO 26262 (automotive) and DO 178C (avionic), require documentation of safety cases for safety-critical systems, and GSN provides the standard format to document the safety cases graphically. The most stable and referenced documentation of GSN is called GSN Community Standard [3]. Some elements of the standard GSN are introduced in Fig. 1.

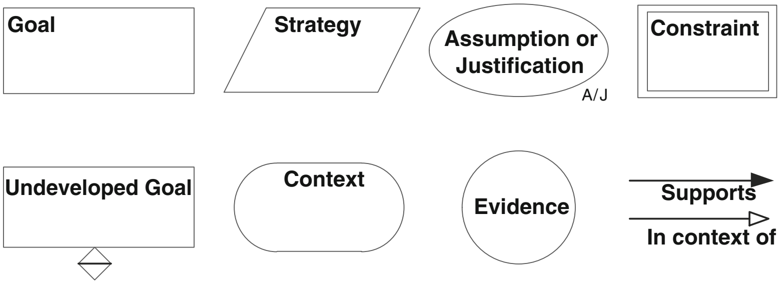


Fig. 1. Some elements of the Goal Structuring Notation.

2.2 Semantics of Business Vocabulary and Rules

Semantics of Business Vocabulary and Rules (SBVR) is a standard business-focused specification proposed by the Object Management Group (OMG) in 2008 [23]. It is designed for domain experts to capture business rules in a formal, structured and understandable language. It consists of a highly flexible structure that can capture and define most of the subtle intricacies of the natural language. The SBVR specification defines, among others, a metamodel to develop models of business vocabulary (comparable to conceptual models) and business rules (comparable to constraints that should be enforced). Figure 2 shows the relevant subset of the SBVR metamodel regarding the concepts presented in this paper. The definitions of some of the main concepts in SBVR specifications [25] are:

Vocabulary, a set of noun concepts, verb concepts, as well as various specialized concepts such as categorizations.
Concept, a unit of knowledge created by a unique combination of characteristics.
Rule, a proposition that is a claim of an obligation or necessity.
Business Rule, a rule with a business focus.

In this paper, all SBVR examples are given in SBVR Structured English (SSE), which is introduced in SBVR Annex C [23]. The font styles used in this paper are shown in Table 1. Note that, for the font style of *Name*, we use the same font style as *Term*.

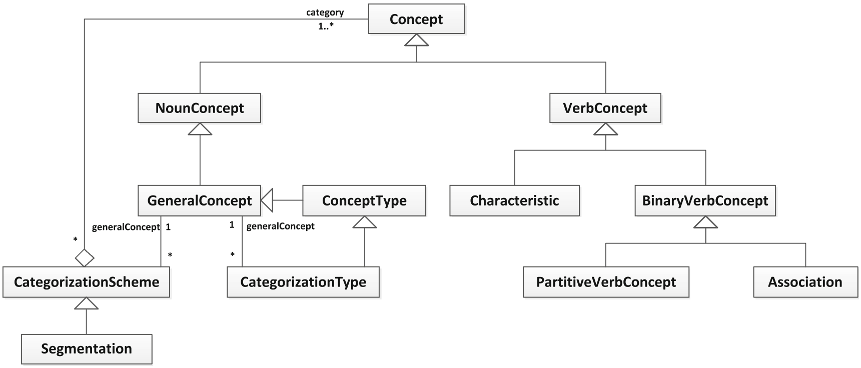


Fig. 2. Extract of the SBVR metamodel.

Table 1. Font styles and color with formal meaning used in this paper.

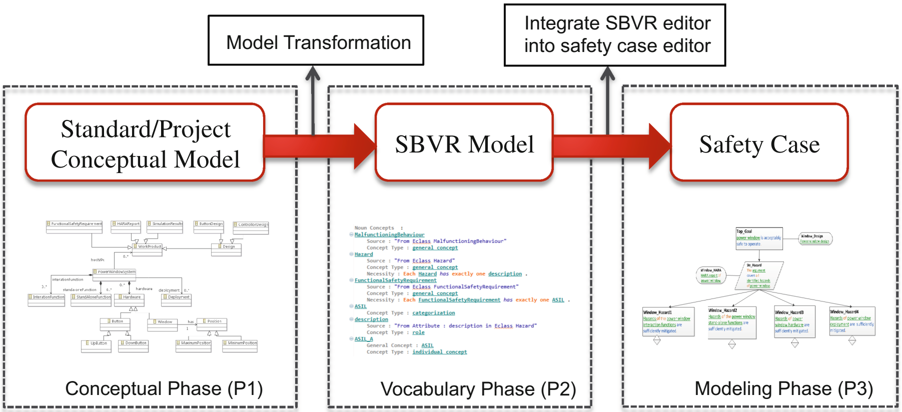
SSE Concepts	Font Style	Color	Denotes
Term	Underlined	Green	Noun concepts or Individual concepts
Verb	Italic	Blue	Verb concepts
Keyword	Normal	Orange	Other linguistic symbols used for definitions and statements

3 Methodology

Based upon our previous approach, we propose a vocabulary-based semi-automatic approach for constructing safety cases. An overview of our current approach is shown in Fig. 3. There are three phases: Conceptual Phase (P1), Vocabulary Phase (P2), and Modeling Phase (P3). In the conceptual phase (P1), a conceptual model of the target domain will be manually built from scratch [19] or semi-automatically refined from other conceptual models [17]. The conceptual model is used as an input for the vocabulary development. The metamodel that

Table 2. Mapping between EMF and SBVR concepts.

EMF concepts	SBVR concepts
Class	General concept
Enumeration literal	Individual concept
Attribute	Role
Association	Verb concept
Generalization/Enumeration	Categorization
Multiplicity	Necessity



```
Noun Concepts  :
MalfunctioningBehaviour
    Source : "From Eclass MalfunctioningBehaviour"
    Concept Type : general concept
Hazard
    Source : "From Eclass Hazard"
    Concept Type : general concept
    Necessity : Each Hazard has exactly one description .
FunctionalSafetyRequirement
    Source : "From Eclass FunctionalSafetyRequirement"
    Concept Type : general concept
    Necessity : Each FunctionalSafetyRequirement has exactly one ASIL .
ASIL
    Concept Type : categorization
description
    Source : "From Attribute : description in Eclass Hazard"
    Concept Type : role
ASIL_A
    General Concept : ASIL
    Concept Type : individual concept
```

Fig. 3. An overview of our methodology.

we use for describing conceptual models is an Ecore metamodel. After the creation of the conceptual model, a model transformation is applied to transform the conceptual model from an EMF format to an SBVR specification. Then in the vocabulary phase (P2), users (argument writers) can build their own vocabulary based on the generated SBVR model. Note that, users can also skip the previous phase and start by creating a new SBVR vocabulary. Finally, in the modeling phase (P3), the vocabulary is used to facilitate safety case construction. The details of these three phases are described in the following subsections.

3.1 Conceptual Phase: From Conceptual Model to SBVR Model

As a preparation phase, the main goal of the conceptual phase is to make the domain knowledge explicit and develop a common understanding. As a result, the development of a conceptual model was proposed in [18]. A conceptual model represents the main terms and their relations that need to be considered for safety cases. As mentioned before, to facilitate the formulation of safety cases, a model transformation is introduced to transform a conceptual model in the EMF format to the SBVR format. The definition of this model transformation is created based on a mapping between the concepts of those two formats (Table 2), and implemented in the Epsilon Transformation Language (ETL) [16]. Finally the conceptual model in SBVR can be used as an input for an SBVR vocabulary in the vocabulary phase. Note that, in our previous research, an SBVR vocabulary can be only built during vocabulary phase.

To demonstrate our current approach and show the improvement on our previous approach, we selected the same example as in the previous study. Figure 4 illustrates a simple EMF schema, which is derived from the industrial ISO 26262 metamodel presented in [26]. It represents the relations between *Malfunctioning Behaviour*, *Hazard*, and *Functional Safety Requirement*. Besides it includes an

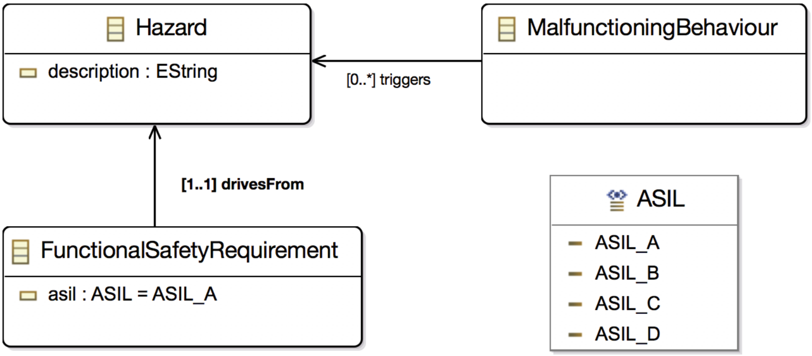


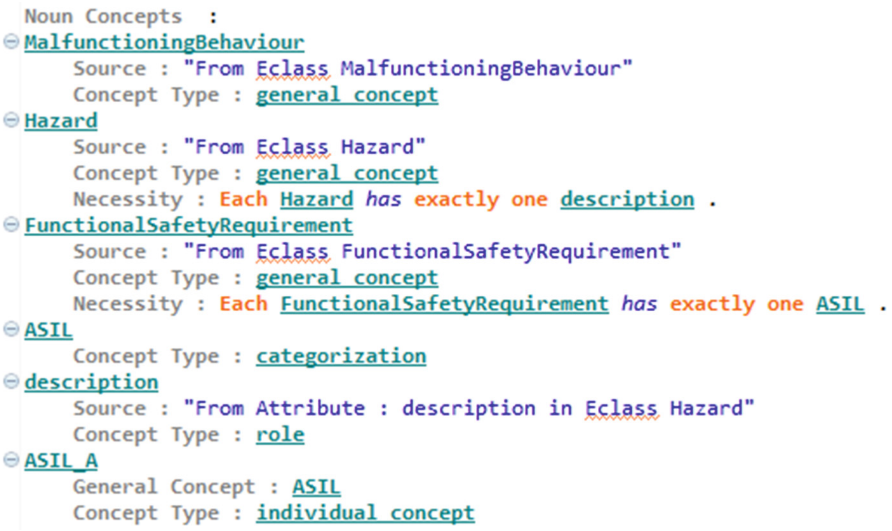
Fig. 4. A part of ISO 26262 metamodel in EMF.

enumeration type, *ASIL*. By applying the model transformation, an SBVR representation of the original conceptual model can be generated. All concepts are categorized into noun concepts and verb concepts. For our example, there are ten noun concepts (three instances of *General Concept*, one instance of *Categorization*, four instances of *Individual Concept*, and two instances of *Role*) and two verb concepts (instances of *Association*) derived from the EMF model. Some of the noun concepts and verb concepts are shown in Figs. 5 and 6 respectively. In these two figures, *Concept Type* represents the type of a noun concept or a verb concept, while *Source* shows the traceability information.

3.2 Vocabulary Phase: Creating Vocabulary with an SBVR Editor

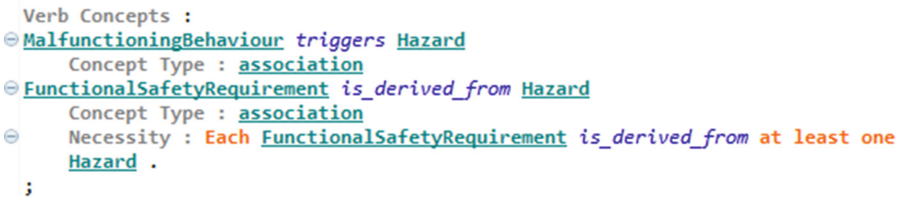
To support safety case development, the generated vocabulary needs to be checked by domain experts or refined with additional safety-related concepts. In this phase, an SBVR editor is developed to support argument writers to build their own vocabulary. As mentioned before, the vocabulary can be a new construction or a modification of an existing one. The detailed information of our SBVR editor will be described in this section.

Main Functions of the SBVR Editor. The main functions of the SBVR editor include a vocabulary editor and a rule editor. The vocabulary editor enables users to define noun and verb concepts, while the rule editor enables users to define SBVR rules with the vocabulary. In our SBVR editor, the "**noun**" font represents general concepts, individual concepts, roles and categorizations whereas the "***verb***" font represents verb concepts. Besides, there are two types of keywords defined in our editor: default keywords and structural keywords. Note that, all keywords are predefined. The default keywords are keywords defined in the SBVR specification. They are displayed in "**keyword**" font style, for example, "**each**" and "**exactly one**". The structural keywords are implemented for the structure of a vocabulary or the characteristics of noun and verb concepts.



```
Noun Concepts  :
MalfunctioningBehaviour
    Source : "From Eclass MalfunctioningBehaviour"
    Concept Type : general concept
Hazard
    Source : "From Eclass Hazard"
    Concept Type : general concept
    Necessity : Each Hazard has exactly one description .
FunctionalSafetyRequirement
    Source : "From Eclass FunctionalSafetyRequirement"
    Concept Type : general concept
    Necessity : Each FunctionalSafetyRequirement has exactly one ASIL .
ASIL
    Concept Type : categorization
description
    Source : "From Attribute : description in Eclass Hazard"
    Concept Type : role
ASIL_A
    General Concept : ASIL
    Concept Type : individual concept
```

Fig. 5. An example of SBVR Noun concepts generated from EMF concepts (screenshot of our editor).



```
Verb Concepts :
MalfunctioningBehaviour triggers Hazard
    Concept Type : association
FunctionalSafetyRequirement is_derived_from Hazard
    Concept Type : association
    Necessity : Each FunctionalSafetyRequirement is_derived_from at least one
    Hazard .
;
```

Fig. 6. An example of SBVR Verb concepts generated from EMF concepts (screenshot of our editor).

They are shown in a gray font style. For instance, in Figs. 5 and 6, "**Noun Concepts**" and "**Verb Concepts**" are defined to categorize concepts into noun or verb. The field of "**Concept Type**" shows the concept type of a noun or verb concept. The field of "**Source**" keeps the traceability information from a EMF model to an SBVR model. Moreover, the field of "**Necessity**" describes the constraints of noun or verb concepts.

There are also predefined noun concepts and verb concepts. The concepts in the SBVR metamodel are predefined noun concepts, such as general concept. The predefined verb concepts are extracted from the SBVR specification to represent some basic association types. The verb concept "***has***" or its passive form "***is_property_of***" identifies the essential properties of a given noun concepts. The containment association is represented by "***includes***" (active form) or "***is_included_in***" (passive form). Besides, the categorization association is represented by "***specializes***", "***generalizes***", "***is_category_of***", or "***is_a***".

According to the SBVR specification, our rule editor supports four types of model operations: obligation formulation, necessity formulation, possibility formulation, and permissibility formulation. A number of keywords are predefined for those model operations. "**it_is_obligatory_that**" and "**it_is_prohibited_that**" are defined for obligation formulation. "**it_is_necessary_that**" and "**it_is_impossible_that**" are defined for necessity formulation. "**it_is_possible_that**" is for possibility formulation. "**it_is_permitted_that**" is for permissibility formulation.

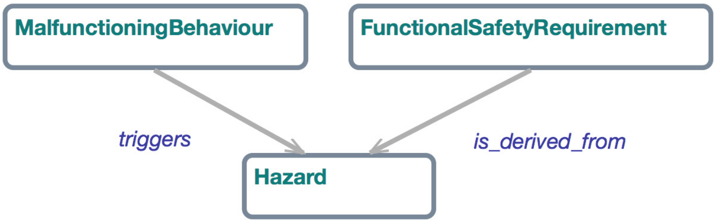


Fig. 7. An illustration of the graphic diagram of verb concepts shown in Fig. 6.

Graphical Editor for Vocabulary. Along with the textual editor, a graphical editor has been implemented to enable users to modify or build their vocabulary via a diagram. A diagram provides an overall picture of a given vocabulary.

All noun concepts and verb concepts defined in the vocabulary are represented in a corresponding diagram. Figure 7 shows a diagram of verb concepts defined in Fig. 6. We can see that, the noun concepts (**MalfunctioningBehaviour**, **Hazard**, and **FuctionalSafetyRequirement**) are shown as nodes, while the verb concepts (***triggers*** and ***is_derived_from***) are shown as links between those nodes. As the textual and graphic editor are synchronized, a noun concept or verb concept can also be added or modified via the diagram. In addition, the properties of concepts can be found in the property view of the graphical element. Labels of noun and verb concepts in the diagram are displayed in the same font color as in the textual editor.

Vocabulary Checking. When building a vocabulary, duplicated noun and verb concepts can be created. This causes a risk of ambiguities and inconsistencies in the vocabulary. Consequently, the confidence of the safety claims, which use those duplicated concepts, will be hampered. To address this, vocabulary checking is implemented in our editor. The goal of the vocabulary checking phase is to check the size of a vocabulary and find duplications in a vocabulary. The output of the vocabulary checking is a report, which shows the number of defined noun and verb concepts and duplicated concepts that need to be addressed. A checking report is automatically generated after a vocabulary created. An example of the checking report is shown in Listing 1.1.

Listing 1.1. An example of vocabulary checking report shows: There are 55 noun concepts defined (line 3–6), 16 verb concepts defined (line 7–10). Besides, there are two noun concpets with the same name "ASIL" (line 11–14).

```
 1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
 2  <reports>
 3     <report>
 4         <check> Definitions of Noun Concepts </check>
 5         <reason> 55 noun concepts are defined </reason>
 6     </report>
 7     <report>
 8         <check> Definitions of Verb Concepts </check>
 9         <reason> 16 verb concepts are defined </reason>
10     </report>
11     <report>
12         <check> Duplicated Concepts </check>
13         <reason> 2 noun concepts with the same name: ASIL </reason>
14     </report>
15  </reports>
```

3.3 Modeling Phase: Construct Safety Cases with Vocabulary

In the modeling phase, the safety argument will be constructed in GSN with vocabulary support. We propose to use SBVR to express the content of each safety case element. By integrating our SBVR editor into the GSN editor, the noun and verb concepts defined in a vocabulary will be highlighted when safety engineers edit a GSN element. Figure 8 shows a screenshot of our GSN editor. When a GSN element is edited, a list of suggested concepts is given via content assistant. For example, after typing "p", a list of concepts in the vocabulary that start with "p" is provided. In this way, the number of errors, such as ambiguities of a safety case can be reduced. Users can always look into the vocabulary to check the definitions of nouns and verbs used in their safety cases to avoid misunderstanding.

4 Case Study

To demonstrate how the controlled language can facilitate safety case construction, we chose two published safety cases for our case study. One is a preliminary safety case for a Whole Airspace Air Traffic Management (ATM) System [1], the other is a safety case for a hypothetical Air Traffic Services Unit (ATSU) [10]. Finally, two main benefits of using controlled language for safety cases are found: the ambiguities in the safety case can be reduced, and the structure of the safety case can be simplified.

Reduce Ambiguities in Safety Case. Figure 9 shows a part of the Whole Airspace ATM system safety case. The concepts or terms which can introduce ambiguities into the safety case are circled. We can see that different terminology is used for expressing the same thing. For example, *Area* and *geographical areas*, *ATM rules* and *Basic ATM rules*, *Assumptions for Area safety* and *safety*

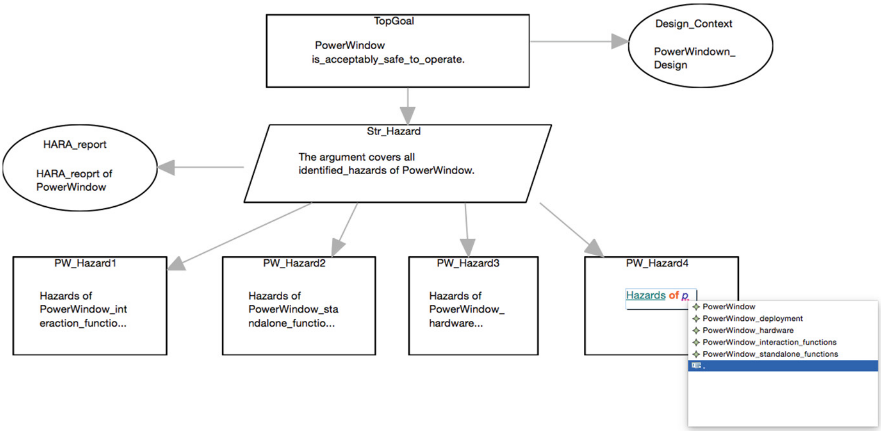


Fig. 8. Illustration of our graphical editor.

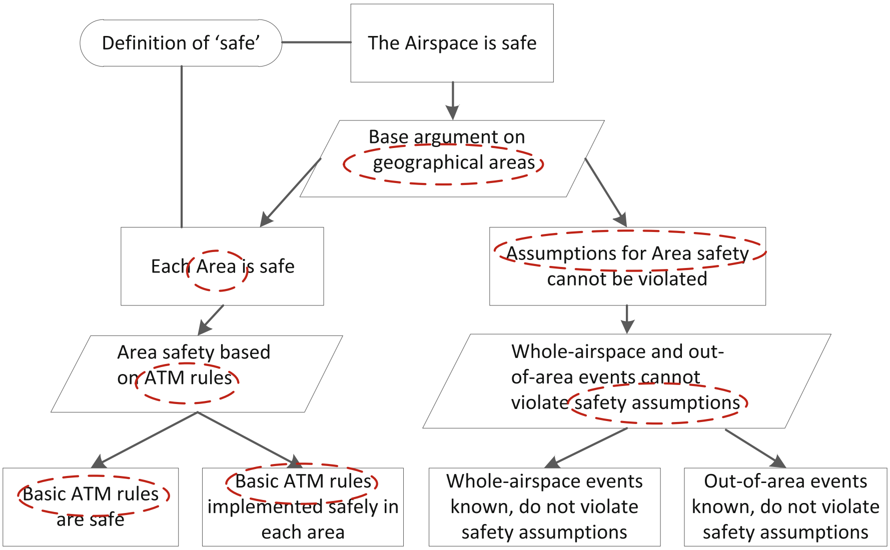


Fig. 9. Extract from 'Complete Structure Based On Geographical Areas Strategy' (Figs. 4, 5 and 6 from [1]).

assumptions. For those similar concepts, some questions can be raised. "Are *geographical areas* a part of or the same as the *Area*?" "Are *Basic ATM rules* a part of or the same as the *ATM rules*?" "Are *Assumptions for Area safety* a part of or the same as the *safety assumptions*?"

If these ambiguities exist in a safety case, the clearness and understandability of the safety case can be affected. Consequently the confidence of the safety case may get harmed. By introducing a controlled language, this issue can be

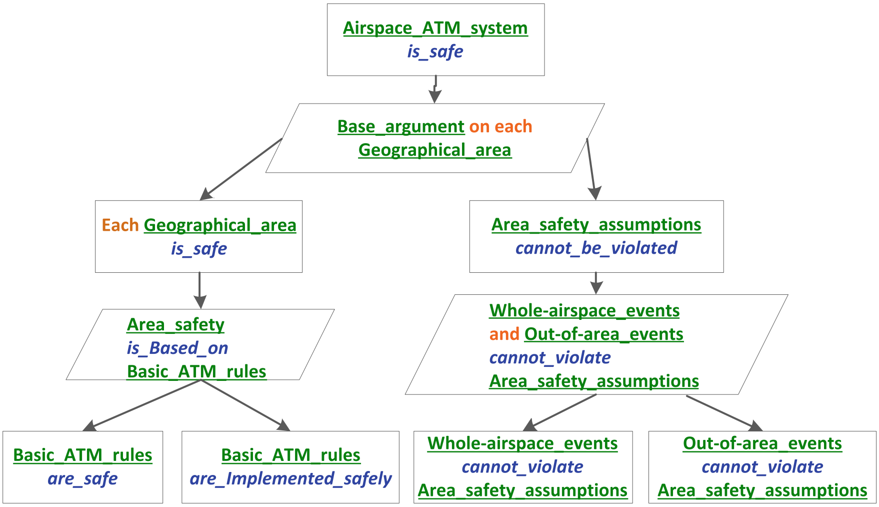


Fig. 10. The SBVR format of the safety case example shown in Fig. 9. The context element *Definition of 'safe'* is removed because the definition of 'safe' can be added in the SBVR vocabulary.

addressed. All the concepts used in the safety case have an explicit definition in the SBVR vocabulary. In this way, the ambiguities in the safety case can be reduced. Figure 10 shows the SBVR format of the previous safety case example (Fig. 9).

Simplify the Structure of Safety Case. Sometimes, in a GSN safety case, safety engineers might add detailed information to the safety case by introducing more GSN elements, for instance a context element. In Fig. 11, we can see that there are three context elements and one constraint element added to a safety claim. These context elements and constraint element can help safety case reviewers to understand the safety claim. However, if the safety case consists of a huge number of safety claims, then elements linked to these safety claims might make the structure of the safety case more complex. Besides, some of these elements are only used to explain the meaning of the terms in a safety claim. For example, in Fig. 11, the context element *C001* and the constraint element *Cr001* are used to provide the definition or guideline for the used terms in the safety claim *Arg0*. By using the controlled language, the information or meaning of terms are provided in the SBVR vocabulary. Therefore, the structure of the safety case can be simplified. Figure 12 shows the SBVR format of the safety case example in Fig. 11. We could see that by moving the definitions of the terms into the SBVR vocabulary, the structure of the safety case gets simplified.

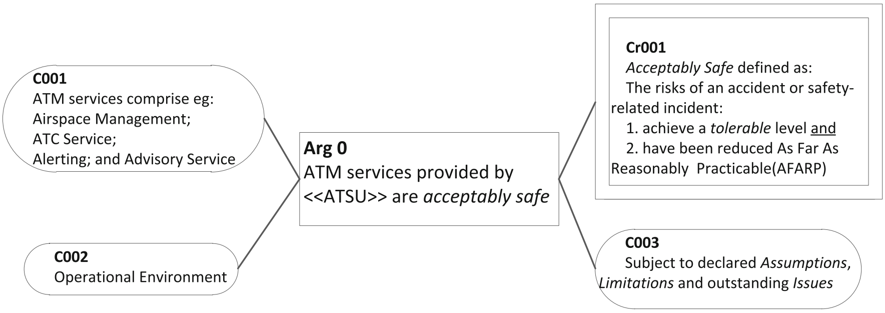

C001
ATM services comprise eg:
Airspace Management;
ATC Service;
Alerting; and Advisory Service
C002
Operational Environment
Arg 0
ATM services provided by <<ATSU>> are acceptably safe
Cr001
Acceptably Safe defined as:
The risks of an accident or safety-related incident:
1. achieve a tolerable level and
2. have been reduced As Far As Reasonably Practicable(AFARP)
C003
Subject to declared Assumptions, Limitations and outstanding Issues

Fig. 11. Extract from 'Overall Safety Argument for a Unit Safety Case' (Fig. 12 from [10]).

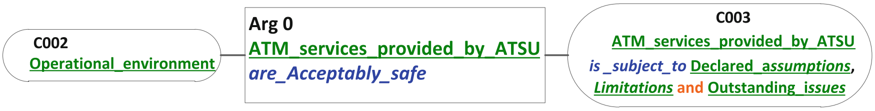

C002
Operational_environment
Arg 0
ATM_services_provided_by_ATSU
are_Acceptably_safe
C003
ATM_services_provided_by_ATSU
is _subject_to Declared_assumptions,
Limitations and Outstanding_issues

Fig. 12. The SBVR format of the safety case example shown in Fig. 11.

5 Related Work

Tool Support for SBVR. In 2006, an SBVR editor called SBeaVeR [8] was developed. SBeaVeR is an Eclipse based plug-in which implements the Structured English notation and provides a number of features: automatic syntax highlighting, automatic completion, dictionary, standard text editing. In 2010, another tool called VeTIS [22], based on the SBVR 1.0 metamodel, was created. Concerning the editing and the validation of SBVR business rules, VeTIS tool provides the same features as SBeaVeR. Both tools provide the main feature for editing SBVR vocabulary, however the vocabulary can only be created from scratch. Whereas in our approach the vocabulary can be derived from a conceptual model.

Tool Support for Safety Case Construction. There exist a number of open source and eclipse-based safety case editors, such as AdvoCATE [9], D-case [20], and CertWare [5]. Besides typical features, they also support safety case modules, patterns, and other advanced functions. In this paper, our focus is on the controlled-language support. Our GSN editor only supports basic features for creating safety cases. However, our vocabulary editor can be integrated into other existing eclipse-based safety case editors. In this way, our approach can be used for other applications.

SBVR and Other Techniques. A model transformation from SBVR to UML with OCL constraints is provided by using ATL (ATL Transformation Language) [22]. Moreover, a study of transforming UML models to SBVR models can also be found in [7]. In this paper, we discussed a model transformation from EMF models to SBVR models.

6 Conclusions and Future Work

In this paper, based on our previous research we presented a vocabulary-based semiautomatic approach for safety case development. By using an SBVR vocabulary, it enables the explicit connection between conceptual models and safety cases to ensure that certification data is built properly and can be reused efficiently. Moreover by utilizing SBVR, the content of safety case elements is well-structured and well-controlled. It can reduce mistakes and misunderstanding between the different roles involved in producing, assessing, and using the safety case. There are four main contributions of this paper:

1. A model transformation from an EMF model to an SBVR model is proposed. A vocabulary can be automatically generated from a conceptual model, which facilities the vocabulary creation.
2. An SBVR editor is implemented. This editor supports the development of SBVR vocabularies and rules, and provides graphical editing and checking of a vocabulary.
3. A GSN editor with SBVR vocabulary support is implemented. This editor enables users to construct safety cases with their own SBVR vocabulary.
4. A case study has been carried out to show the benefits of using controlled language for safety case construction. The results show that by using controlled language, the ambiguities in the safety case can be reduced and the structure of the safety case can be simplified.

Thus, our method supports improving the clarity and correctness of safety cases, and increasing the confidence in the claimed safety assurance. As future work, we plan to extend our approach on safety case pattern development for reuse of certification data. Finally, we aim to cooperate with industrial partners to use our approach in large scale applications.

Acknowledgements. The research leading to these results has received funding from the FP7 programme under grant agreement n° 289011 (OPENCOSS).

References

1. EUR Whole Airspace ATM System Safety Case (2001). http://dependability.cs.virginia.edu/research/safetycases/EUR_WholeAirspace.pdf
2. Tube Lines Contractual Safety Case (2004). http://dependability.cs.virginia.edu/research/safetycases/Tube_Lines.pdf
3. GSN Community Standard Version 1 (2011). http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf
4. Software Consideration in Airborne Systems and Equipment Certification: RTCA DO-178C, December 2011
5. Barry, M.R.: CertWare: a workbench for safety case production and analysis (2011)
6. Bishop, P., Bloomfield, R.: A methdology for safety case development. In: Redmill, F., Anderson, T. (eds.) Industrial Perspectives of Safety-Critical Systems, pp. 194–203. Springer, London (1998)

7. Cabot, J., Pau, R., Raventós, R.: From UML/OCL to SBVR specifications: a challenging transformation. Inf. Syst. **35**(4), 417–440 (2010)
8. De Tommasi, M., Corallo, A.: SBEAVER: a tool for modeling business vocabularies and business rules. In: Gabrys, B., Howlett, R.J., Jain, L.C. (eds.) KES 2006. LNCS (LNAI), vol. 4253, pp. 1083–1091. Springer, Heidelberg (2006)
9. Denney, E., Pai, G., Pohl, J.: AdvoCATE: an assurance case automation toolset. In: Ortmeier, F., Daniel, P. (eds.) SAFECOMP Workshops 2012. LNCS, vol. 7613, pp. 8–21. Springer, Heidelberg (2012)
10. EUROCONTROL: Safety Case Development Manual (2006). http://www.eurocontrol.int/sites/default/files/article/content/documents/nm/link2000/safety-case-development-manual-v2.2-ri-13nov06.pdf
11. Graydon, P.J.: Towards a clearer understanding of context and its role in assurance argument confidence. In: Bondavalli, A., Di Giandomenico, F. (eds.) SAFECOMP 2014. LNCS, vol. 8666, pp. 139–154. Springer, Heidelberg (2014)
12. Hawkins, R., Kelly, T., Knight, J., Graydon, P.: A new approach to creating clear safety arguments. In: Dale, A., Anderson, T. (eds.) Advances in Systems Safety, pp. 3–23. Springer, London (2011)
13. ISO: ISO 26262: "Road Vehicles - Functional Safety" (2011)
14. Kelly, T.: Arguing Safety - A Systematic Approach to Managing Safety Cases. Ph.D. thesis, University Of York (1998)
15. Kelly, T., Weaver, R.: The goal structuring notation - a safety argument notation. In: Proceedings of Dependable Systems and Networks 2004 Workshop on Assurance Cases (2004)
16. Kolovos, D.S., Paige, R.F., Polack, F.A.C.: The epsilon transformation language. In: Vallecillo, A., Gray, J., Pierantonio, A. (eds.) ICMT 2008. LNCS, vol. 5063, pp. 46–60. Springer, Heidelberg (2008)
17. Luo, Y., van den Brand, M., Engelen, L., Klabbers, M.: From conceptual models to safety assurance. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) ER 2014. LNCS, vol. 8824, pp. 195–208. Springer, Heidelberg (2014)
18. Luo, Y., van den Brand, M., Engelen, L., Klabbers, M.: A modeling approach to support safety assurance in the automotive domain. In: Selvaraj, H., Zydek, D., Chmaj, G. (eds.) Progress in Systems Engineering. Advances in Intelligent Systems and Computing, vol. 330, pp. 339–345. Springer International Publishing, Switzerland (2015)
19. Luo, Y., van den Brand, M., Engelen, L., Favaro, J., Klabbers, M., Sartori, G.: Extracting models from ISO 26262 for reusable safety assurance. In: Favaro, J., Morisio, M. (eds.) ICSR 2013. LNCS, vol. 7925, pp. 192–207. Springer, Heidelberg (2013)
20. Matsuno, Y.: D-Case Editor: A Typed Assurance Case Editor. University of Tokyo (2011)
21. Defence Standard 00-55 Part 1 (1997). http://www.software-supportability.org/Docs/00-55_Part_1.pdf
22. Nemuraite, L., Skersys, T., Sukys, A., Sinkevicius, E., Ablonskis, L.: VeTIS Tool for Editing and Transforming SBVR Business Vocabularies and Business Rules into UML & OCL Models (2010)
23. OMG: SBVR: Semantics Of Business Vocabulary And Rules (version 1.2), Sepetember 2013
24. Safety Case Repository (2013). http://dependability.cs.virginia.edu/info/Safety_Cases:Repository

25. Spreeuwenberg, S., Healy, K.A.: SBVR's approach to controlled natural language. In: Fuchs, N.E. (ed.) CNL 2009. LNCS, vol. 5972, pp. 155–169. Springer, Heidelberg (2010)
26. Taguchi, K.: Meta Modeling Approach to Safety Standard for Consumer Devices (2013)