Automated Evidence Analysis of Safety Arguments using Digital Dependability Identities

Jan Reich[1], Marc Zeller[2], and Daniel Schneider[1]

[1] Fraunhofer IESE, Kaiserslautern, Germany
{jan.reich,daniel.schneider}@iese.fraunhofer.de
[2] Siemens AG, Munich, Germany
marc.zeller@siemens.com

Abstract. Creating a sound argumentation of why a system is sufficiently safe is a major part of the assurance process. Today, compiling a safety case and maintaining its validity after changes are time-consuming manual work performed by safety experts based on their experience and knowledge. This work is further complicated when supplier components need to be integrated where important details might not be known. By using the concept provided by *Digital Dependability Identities (DDI)*, we present an approach to automatically check evidence validity for safety requirements through leveraging from formal traceability between safety argument and evidence models being both parts of the DDI. This approach reduces the effort for creating and maintaining the system-level safety argument by (a) performing automated evidence analysis for safety requirements, (b) supporting a model-based multi-tier safety engineering process and (c) eliminating the human error source by relying on DDI scripts to encode safety engineering activities. We illustrate our approach using a case study from the railway domain, which focuses on the safety assurance of a train control system (ETCS).

1 Introduction

The growing complexity of safety-critical systems in many application domains such as the automotive, avionics, or railway poses new challenges for system development. Along with the growing system complexity, also the need for safety assurance and its associated effort is drastically increasing. Safety assurance is a mandatory part in order to pass certification and to satisfy the high-quality demands imposed by the market. Consequently, without safety assurance, market introduction and success of a product is in jeopardy. In different application domains, safety standards such as ISO 26262 [3] in the automotive domain or CENELEC EN 50129 [2] for railway signaling equipment define the safety assurance process. The goal of this process is to identify all failures that cause hazardous situations and to demonstrate that their occurrence probabilities are sufficiently low and that the system's residual risk is thus acceptable. Typical

activities in this process are the identification and assessment of hazards, conducting safety analyses (e.g. using FTA) and eventually the creation of a sound safety argument as to why the system is sufficiently safe. The safety argument clearly has a central role as it communicates the relationship between safety requirements and its supporting evidence [5]. The argumentation is often part of a safety case (depending on the domain) and its creation can generally be considered a major task of the overall assurance process.

Compiling a safety case is typically very laborious and, as of today, is purely manual work performed by safety experts based on their experience and knowledge about the applied safety standard and the system under consideration. There is a lack of guidance, general tool-support and, not surprisingly, there is no automation available to support the argumentation and the linking of evidence to create the rationale for why a system is sufficiently safe.

By using the concepts provided by *Digital Dependability Identities (DDIs)* [10], we present an approach to automatically check the sufficiency of evidence for safety requirements by automating parts of the assurance process. Particularly, by means of the presented approach, the effort of compiling a sound safety case is reduced by automating the creation of the safety argumentation about the development process and the product itself. Likewise, the effort for the reassessment of a safety-critical system after performing a modification can be decreased. The presented approach further supports different engineering scenarios in the context of a multi-tier engineering process, i.e. the synthesis of modular safety-related specifications of a component (in form of a DDI) by a supplier as well as semi-automated support for the integration of different DDIs (of supplier components) by an OEM. We illustrate our approach using a case study from the railway domain, which focuses on the safety assurance of a train control system, i.e. the *European Train Control System (ETCS)*, as a running example.

The rest of the paper is organized as follows: In Sec. 2, we introduce the railway case study which is used as a running example. Afterward, we outline the concept of DDIs for safety engineering and show how it is embedded in a distributed development process. In Sec. 4, we present our approach to automate the creation of a safety argumentation about both processes and product. The state-of-the-art is briefly summarized in Sec. 5. Sec. 6 concludes the paper.

2 ETCS Running Example

2.1 ETCS System Description

The *European Train Control System (ETCS)* provides standardized train control in Europe and facilitates cross-border train operation. ETCS consists of onboard and trackside subsystems. Moreover, the ETCS subsystems may be produced by different vendors (suppliers) and must be integrated by a railway operator. Both subsystems must fulfill the safety requirements defined in Subset-091 of the ERTMS/ETCS specification [11]. However, functions such as the emergency brake service consist partly of on-board as well as parts of trackside functionalities. Thus, a proper analysis of the emergency brake function must span over

both sub-systems. Consequently, the safety case of an ETCS system needed for certification w.r.t. CENELEC EN 50129 [2] requires the synthesis of a sound safety argument by the system integrator using the information provided by the different subsystem suppliers.

2.2 ETCS Safety Case

This section describes an exemplary safety case related to the ETCS trackside system. The goal of a safety case is to demonstrate in a structured argument with evidence that a system's residual safety risk has been reduced to an acceptable level. In many application domains, there are already standards in place providing a set of processes or concrete safety requirements to be executed and satisfied, depending on the required integrity. Thus, safety standards such as CENELEC EN 50129 provide a basis to structure the high-level safety argument. Fig. 1 shows those parts of an exemplary ETCS safety case relevant for the paper, documented in the Goal Structuring Notation (GSN) [6]. The refinement of the top-level Goal G1 is on the one hand driven by adherence to railway safety standards as well as to the ETCS system safety specification. On the other hand, it is driven by the system decomposition into trackside and onboard subsystems, of which the trackside system will be the focus of this paper (see Fig. 1). Thereby, the root safety goal is refined into process and product argument parts.

The process-related requirements typically directly originate from safety standards implying required process execution rigor according to the Safety Integrity Level (SIL). As (at least some) trackside functions are determined highly safety-critical, they have to be developed demonstrating SIL 4 integrity (G2). In Fig. 1, G4, G7, G8 and G9 exemplify such normative requirements, of which the automated validity analysis of G9 will be elaborated in Sec. 4.3.

As rigorous process execution is only an indirect and therefore not definitive measure of safety, it has to be complemented by product-related safety requirements constraining failure rates with respect to system-level hazardous events to an acceptable level (G3 in Fig. 1). The acceptable failure rate thresholds for the hazards of the ETCS system are defined in the ETCS specification Subset-091 [11].

It is important to note that different parts of the safety case are created and assessed by different stakeholders. While the overall system part of the safety argument is the responsibility of the ETCS system integrator, G1, G2 and G3 represent the development interface with the Trackside system supplier, i.e. their further decomposition and the provision of evidence is the supplier's responsibility. This enables the supplier to be flexible regarding the choice of safety analysis techniques to demonstrate that the target failure rate has been achieved by the supplied system. For providing the required evidence to support satisfaction of trackside system goals G3, G5 and G6, Component Fault Trees (CFTs) [4] have been created. Therefore, G5 and G6 can be verified by performing a quantitative Fault Tree Analysis (FTA) of the top events representing the hazards *Erroneous balise telegram interpretable as correct* and *Erroneous loop telegram interpretable*

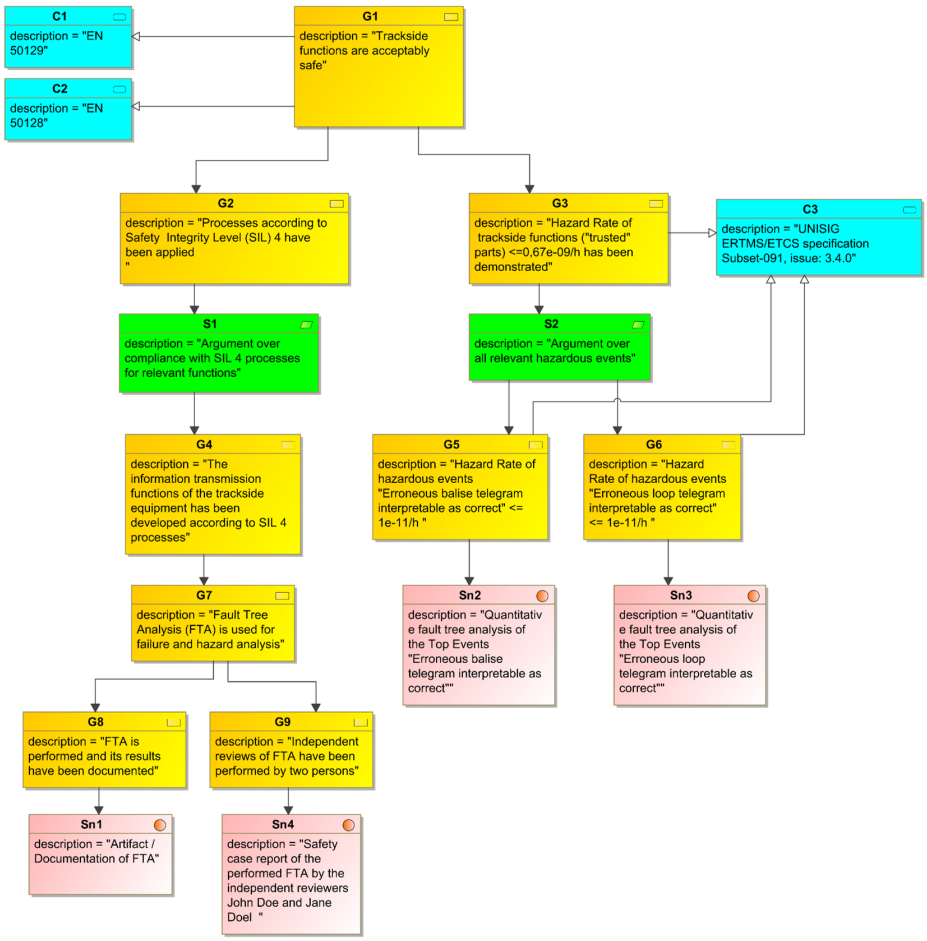

Fig. 1. Exemplary ETCS safety case

as correct. In addition, G4 and its sub goals as the related process requirements make sure that the CFT creation and maintenance processes yield an adequate CFT model so that the CFT analysis results can be trusted. Typically, the evidence for adequate safety process adherence is found in process handbooks, review sheets or configuration management tools.

3 Distributed Safety Engineering with DDIs

This section describes the general concept of Digital Dependability Identities and how they can be used to perform continuous safety engineering processes in different integrator-supplier scenarios.

3.1 Digital Dependability Identities (DDIs)

A fundamental problem of current dependability engineering processes hampering effective assurance lies in the fact that safety argument models are not formally related to the evidence models supporting the claim. Concrete examples for such evidence are hazard and safety analysis models or dependability process execution documentation. As all those artifacts refer to the same system and therefore are naturally interrelated with each other, we claim this should also be the case for the systems model-based reflection: The Digital Dependability Identity (DDI) [10]. By establishing this kind of traceability, DDIs represent an integrated set of dependability data models (=What is the evidence data?) that are generated by engineers and are reasoned upon in dependability arguments (=How is the evidence data supporting the claim?). A DDI (see 2) is, therefore, an evolution of classical modular dependability assurance models, in that several separately defined dependability aspect models are now formally integrated allowing for comprehensive dependability reasoning. DDIs are produced during design, certified when the component or system is released, and then continuously maintained over the lifetime of a component or system. DDIs are used for dependable integration of components into systems during development.

A DDI contains information that uniquely describes the dependability characteristics of a system or component. DDIs are formed as modular assurance cases, are composable and can be synthesized to create more complex DDIs from the DDIs of constituent systems and system components. The DDI of a system contains a) claims about the dependability guarantees given by a system to other systems b) supporting evidence for the claims in the form of various models and analyses and c) demands from other connected systems being necessary to support the claims.

The starting point for all dependability assurance activities is the description and planning of the functionality that the system shall render for its stakeholders, which may be either direct system users, companies or even the society. An essential property of a function is that it is executed on multiple independently developed subsystems leading to a required distribution of dependability assur-

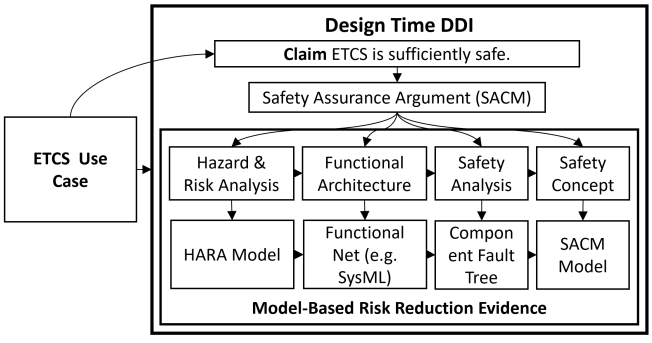

Fig. 2. Digital Dependability Identity @ Design Time

ance over multiple system manufacturers. For example, the ETCS function is executed on the train's onboard system and the trackside system. Enabling cooperative function execution while still allowing decoupled development is only possible by making development interfaces explicit for both functional and quality aspects. Concretely, structural and behavioral aspects of the intended system function need to be made explicit along with assured constraints regarding their quality bounds.

DDIs are concerned with the comprehensive and transparent assurance of dependability claims. Thus, each assurance activity and each artifact contained in a DDI is motivated by a root dependability claim that defines sufficient risk reduction regarding a dependability property such as safety, security, availability or reliability. The definition of acceptable risk reduction is typically derived from domain-specific risk management standards targeting different risk causes such as functional safety (e.g. CENELEC EN 50129). These standards contain requirements for assessing and reducing risks to an acceptable level.

Having a dependability claim to be assured for the system function, the next step is the systematic planning of risk management activities. These activities create necessary evidence for supporting the system engineers reasoning that the dependability claim holds for the developed system. For both risk management planning and dependability assessment purposes, an explicit argument is indispensable inductively relating the created evidence to the top-level claim through several step-wise layers of argumentation. Note that, while the performed activities and produced artifacts vary depending on the kind of risk that is being managed, the general need for argumentation supported by evidence is mandatory for all risks. DDIs deal with dependability risks, thus the currently supported design time DDI assurance activities and evidence focus on well-established dependability methods such as hazard and risk analysis, safety and security analyses, safety design concepts, validation, and verification. These activities proved sufficient over the last decades in demonstrating the dependability of embedded systems. In addition, the reliance on model-based approaches compensated for the increasing complexity of systems in the past.

Fig. 2 illustrates the concept of continuous traceability between a SACM safety argument and safety-related evidence models stemming from hazard and risk analysis, functional architecture, safety analysis, and safety design concept. SACM stands for *Structured Assurance Case Metamodel* and was standardized by the OMG on the basis of GSN. [7]. It provides the assurance case backbone for creating the required traceability. Apart from relating evidence models formally to the assurance argument, a unique contribution of DDIs is the concretization of semantics specifically for safety assurance evidence. Based on this added product semantic, safety engineering activities in form of *DDI scripts* can be executed on the DDI data contents automatically. The DDI meta-model formalizing the described traceability and evidence semantics is the *Open Dependability Exchange (ODE)*[3] meta-model. Although we describe DDI usage in this paper for safety assurance activities, the concept is general enough to equally apply it to other

[3] see http://www.deis-project.eu/ and https://github.com/DEIS-Project-EU/

dependability properties such as security, reliability or availability. More details on the DDI framework, its technical realization and usage benefits can be found on the DEIS project website's dissemination section [1].

3.2 Generic safety engineering process integrator/supplier

In order to be able to identify engineering tasks that are supported by partial or full automation, it is required to anticipate a certain development process in which the engineering tasks are embedded. Fig. 3 shows an abstract development process that is representative of domains such as railway or automotive. There is an integrator company (e.g. the OEM or the railway operator) that is building a system by integrating a set of components that are provided by supplier companies. This process involves four steps, in which the DDI concept together with (semi-)automated engineering support lead to improvement.

Step 1 - DDI Synthesis @ Integrator Step 1 involves the synthesis of component specifications that the supplier company must adhere to. One particular challenge of synthesizing this specification is to collect all relevant information that is needed by the supplier in order to develop the component in isolation. This is not only necessary for information about required functionality, but also for safety requirements. In this scenario, DDIs can be seen as a container, where all this information can be captured in an integrated and structured way. Thus, engineering tool support for this step should focus on helping the engineer to collect the relevant information required by the DDI for the specific tasks the supplier should carry out. Such specific tasks could be for instance to demonstrate the adequate satisfaction of interface safety requirements or to check the compatibility of the supplier component interface with the interface definition provided by the integrator.

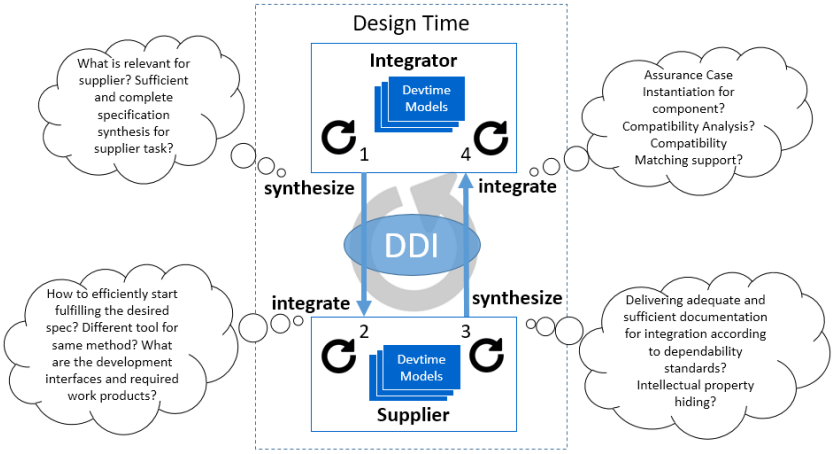

Fig. 3. Generic distributed engineering process of safety-critical systems

Step 2 - DDI Integration @ Supplier Step 2 represents the integration of the specification DDI into the development process of the supplier company. This could mean for example that model stubs are automatically generated based on the development interface extracted from the imported DDI. Such a development interface typically consists of functional or technical data-flow interfaces or safety requirements allocated to those data-flow interfaces. This could also be placeholders for assurance evidence artifacts that are to be instantiated by the supplier and connected to the actually created evidence in shape of safety analysis, verification, validation or architectural models.

Step 3 - DDI Synthesis @ Supplier After the DDI has been integrated in a (semi-)automated way, the supplier performs the actual development work until it is time to deliver the component back to the integrator. In this instance, delivery means not only the physical component but also the safety-related documentation that is required to build a sound safety case for the integrated system (typically according to one of the commonly known standards). Step 3 is concerned with synthesizing a DDI containing all relevant information that is needed so that the integrator can properly perform the integration task. From the supplier perspective, all *relevant* information means explicitly not all *existent* information and, therefore, engineering support for the supplier should focus on identifying and collecting the minimal set of information to be delivered. Although this paper focuses on the integrator perspective, the presented DDI approach could in principle also be used by the supplier to automatically check component safety requirement validity through checking traceability from requirements to software/hardware design to code and their respective validation activities such as tests.

Step 4 - DDI Integration @ Integrator Step 4 deals with the integration of a component DDI into the overall system. This engineering task typically involves performing compatibility and structural analyses as well as behavioral compatibility matching with other components. This task needs to be performed for both functional and safety aspects. Thus, the component safety case has to be integrated into the system safety case in order to demonstrate confidence in the high-level safety assurance claims. Engineering support should focus on the (semi-)automated generation, integration and verification of safety case fragments, i.e. (semi-)automated assessment of claim satisfaction in system safety cases. Thereby, it is necessary to assess whether the provided argumentation and evidence yields sufficient confidence in the validity of system-level assurance claims. In order to assess the adequacy of evidence and argumentation with partial or full automation, we make use of the aforementioned formal interrelation of safety case models and product models provided by a DDI. This approach enables assessing how changes in any of the DDI product models propagate up to the safety case and indicating to the dependability engineer the impact of changes on the validity of claims. As safety cases and the models representing the evidence for supporting the claims tend to be quite large in the real world, partially automated change impact analysis gives valuable information to the engineer, which parts of the existing safety artifacts need to be reassessed.

4 DDI-based Evidence Analysis in a Railway Case Study

This section first describes the ETCS System DDI (Sec. 4.1) and afterward exemplifies the DDI engineering approach for two concrete safety engineering activities related to the verification of product-related (Sec. 4.2) and process-related (Sec. 4.3) safety requirements.

4.1 ETCS System DDI

In the ETCS use case, we utilize DDIs to represent the information provided in the safety assurance process of the ETCS system, in particular, to turn the integration of the trackside subsystem safety assurance artifacts more efficient.

A high-level overview of a DDI and its contents for the ETCS example is depicted in Fig. 4. The ETCS DDI's backbone is the system-level safety argument expressed in SACM. As described in Sec. 2.2, the concrete safety argument contains a process-related part motivated by safety requirements from CENELEC EN 50129 and a product-related part mostly driven by the ETCS specification. In hierarchical system (of system) structures, the refinement of the system-level safety argument results in safety requirements to be satisfied by the subsystems, in the ETCS case by the trackside and onboard subsystems. Note that the principal structure of the trackside system DDI (lower part of Fig. 4) is almost equal to the ETCS DDI, with the exception that the root (=the safety guarantees to be given) of the trackside safety case are the interface safety requirements posed by the ETCS integrator. From that point on, a safety argument needs to be provided by the trackside system manufacturer within the context of the trackside system. This safety argument is supported by evidence artifacts synthesized from various kinds of models such as failure logic, architecture or process models. The most notable innovation introduced by DDIs is that the source models for evidence are formally linked to the argument bits supported by them and organized within an all-embracing container. This characteristic together with the possibility to automatically match demanded and satisfied requirements in different DDIs (see the "Trackside System Safety Requirements" *demanded* by the ETCS system and *satisfied* by the trackside system) enable efficient safety-related collaboration across multi-tier supply chains and semi-automated change management through explicitly defined exchange interfaces. After the DDIs of all subsystems have been integrated into the ETCS system DDI by using DDI scripts, the last step in the assurance process is the provision of integration-level evidence that is typically based on analyses of the integrated system's architecture and failure logic as well as the verification and validation of safety goals and that any assumptions posed during the development process still hold in the final product.

4.2 Automated Evidence Analysis of Product Safety Argument

In this Section, we want to show how to automatically analyze evidence completeness for the following ETCS product safety requirement across the supply

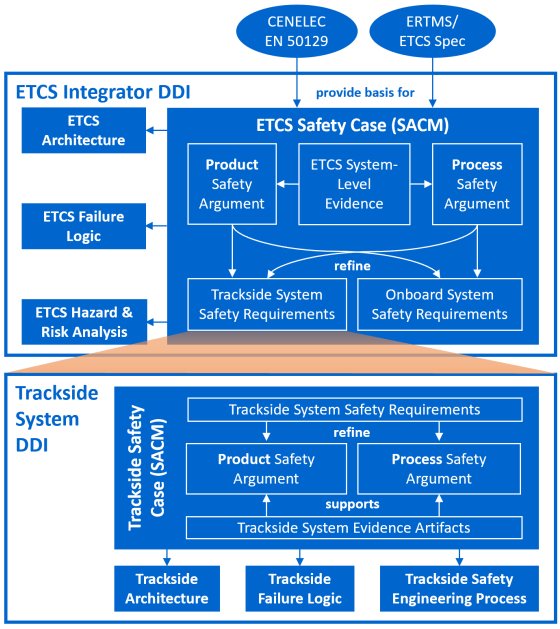

Fig. 4. High-level contents of ETCS and trackside DDIs

chain with the DDI framework (see Figures 1 and 5).
G3: *Hazard rate of trackside functions ("Trusted part")* $\leq 0.67 \cdot 10^{-9}/h$ *has been demonstrated.*
In order to analyze requirement G3 in a conventional safety assurance process (i.e. without DDIs), the ETCS *integration* safety engineer would have to perform the following steps:

S1 Determine, which activities and work products a standard-compliant satisfaction of the requirement implies
S2 Refine G3 according to the assurance strategy into interface safety requirements for all hazardous events (G5 and G6 as examples)
S3 Contact the *trackside* system's safety engineer and get documentation (e.g. documents or models) about evidence for the satisfaction of G5 and G6
S4 Identify the relevant parts within the provided documentation, in this case, the FTA models and analysis results (typically time-consuming due to different processes, tools and safety assurance perspectives)
S5 Sum up the supplier quantitative FTA results for the relevant hazardous events ($2 \cdot 10^{-11}/h$) and compare the result to the target failure rate demanded by G3 ($0.67 \cdot 10^{-9}/h$).
S6 If the requirement is fulfilled, link and archive evidence source to the requirement so that an assessor can find it easily.

Fig. 5 shows how the engineering steps to verify requirement G3 listed above can be carried out automatically with the DDI framework. The starting points

are the safety requirements G5 and G6, which are located in SACM Argument packages of *both* ETCS and trackside DDIs, in the first as part of the argument refinement, in the latter as explicit safety interface specification (in Fig. 5, duplication was avoided). Thus, evidence produced for the trackside system can be traced to from the ETCS safety argument during integration by automatically matching. This automatic matching is currently done based on tagged value content matching. However, more intelligent matching mechanisms such as semantic understanding of structured requirements and evidence descriptions are to be explored. Having established traceability between ETCS and trackside arguments, the main question is how to (a) relate G5 with the failure logic model supporting the requirement and (b) how to perform steps S4-S6 automatically, i.e. how to use the enriched data to produce the verification result.

To support the understanding of a requirement by a machine, SACM provides the *Terminology* mechanism, which enables structured text definition. In the example this means that G5 is enriched with semantic tags (called *Terms*) representing the nature of the requirement:

RequirementType The type of demand (=Failure Rate Demonstration with Quantitative FTA), related to a DDI script encoding a safety standard ac-

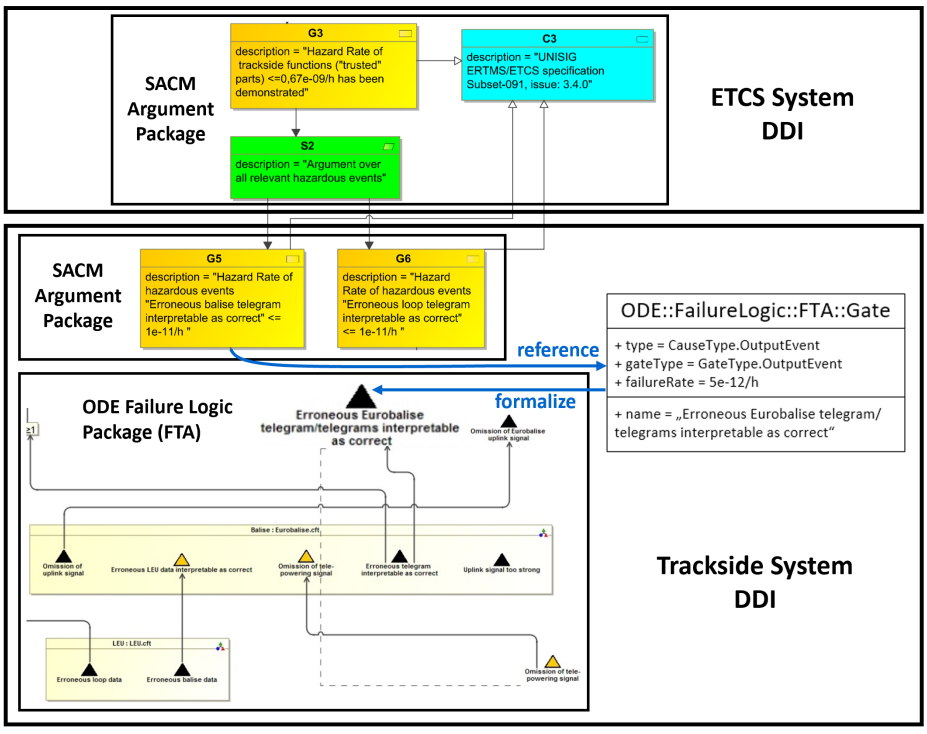

Fig. 5. ETCS Failure Rate Consistency Check with DDIs

tivity consuming all required DDI models and producing the verification result.

ElementReference Formal reference to the TopEvent element in the Trackside CFT representing the hazardous event described in the requirement. In general, this can be any element within the DDI.

VerificationCriteria The target failure rate to be satisfied by the actual computed failure rate for the fault tree (assumed to be existent in the failureRate attribute of the top events ODE representation). In general, these criteria should be defined with Boolean conditions to ensure decidability.

Based on a fully integrated DDI data model containing all required verification information, the missing piece is the encoded safety intelligence performing the automated verification. For this task, a script-based safety automation engine has been created in *DEIS* relying on the Eclipse Epsilon Framework [4]. It offers various task-specific languages for efficiently querying, manipulation and generation of models. For the ETCS verification task, DDI scripts have been created operating on the DDI models shown in Fig. 5. To that end, the *Epsilon Validation Language (EVL)* and the *Epsilon Object Language (EOL)* have been employed to navigate within and between the DDIs, generate new information such as analyzing the CFT (EOL) and checking structural constraints such as the verification criteria on it (EVL). Note that this paper's focus is on the conceptual idea of using DDIs for the evidence completeness analysis of different kinds of safety requirements. We plan to elaborate on its technical realization with the *Epsilon Framework* in a further publication. However, in our project deliverables [1], more details can be found.

For the ETCS example, the DDI script for checking failure rate satisfaction (G5) performs the following tasks:

1. Retrieve those claims in the SACM safety argument representing quantitative analysis requirements (indicated by the *RequirementType* Term).
2. For each identified claim, locate the referenced top event element in the FailureLogicPackage of the subsystem DDI and retrieve the actual failureRate attribute. If non-existent at that point, invoke another DDI script to perform an on-demand quantitative FTA for the component fault tree related to the top event.
3. Compare the actual failure rate with a target failure rate and report a violation to the safety engineer if the actual failure rate is higher than the actual failure rate.

The DDI script content outlined above is able to analyze the satisfaction of single budgeted failure rate requirements. Finally, summing up the actual failure rates of multiple contributing (sub)system parts (G3) is done in a separate DDI script orchestrating the *Failure Rate Demonstration Scripts* for all relevant contributing parts. This demonstrates, how higher-level requirement satisfaction can be analyzed by reuse of lower-level safety engineering DDI scripts.

[4] https://www.eclipse.org/epsilon/

4.3 Automated Evidence Analysis of Process Safety Argument

For process-related safety requirements like G8 and G9 stated below, the engineering activities for requirement verification are similar to those described for product-related requirements in the previous section. The only difference is that the queried DDI contents are referring to model-based process execution documentation formally related to the safety artifacts (i.e. CFT models) produced by the respective process (i.e. standard-compliant FTA).
G8: *FTA is performed and its results have been documented.*
G9: *Independent reviews of FTA have been performed by two persons.* Fig. 6 shows the relevant trackside DDI contents for verifying the above requirements resulting from ETCS process argument refinement. Evidence related to process execution can be captured in SACM *Artifact* packages along with the artifacts the process relates to. Therefore, *Events* and *Activities* including their *Participants* as well as used *Resources* and *Techniques* can be tracked for an *Artifact*. Within the DDI, SACM *Artifacts* may be formally connected to the product safety models, e.g. the CFT in the DDI's failure logic package.

In order to automatically verify the evidence of G8 and G9 the following logic has been codified in EVL DDI scripts:

G8 The FTA Artifact related to G8 should be associated with (a) the actual produced fault tree models as part of the DDI's failure logic package and (b) the Activity Hazard or Failure analysis, where the used Technique is FTA and the used resource is the trackside system architecture referencing the DDI's trackside system architecture package including the safety-relevant trackside functions.

G9 The FTA Artifact related to G9 should be associated with (a) the actual produced fault tree models as part of the DDI's failure logic package and (b) an Event Review, where there are at least two associated Participants being different people.

5 Related Work

Related work can be broadly categorized into the two different areas a) the formalization of specific safety aspects into meta-models usable for automation and b) the integration of multiple aspects for more comprehensive reasoning. Regarding a), common techniques have been formalized for the definition of system failure propagation (e.g. Component Fault Trees [4], HiP-HOPS [8]) and the corresponding safety argumentation supported by evidences (GSN, SACM) [5, 7]. Using these techniques separately in different phases of the engineering process proved to be very valuable in the past, but their lack of integration with each other makes proper reaction to changes a nearly impossible task. Therefore, some initiatives aim at integrating safety aspects to allow more extensive safety reasoning supported by automation. In the SPES project series [9], the *Open Safety Metamodel* has been created enabling a modular, cross-tool and

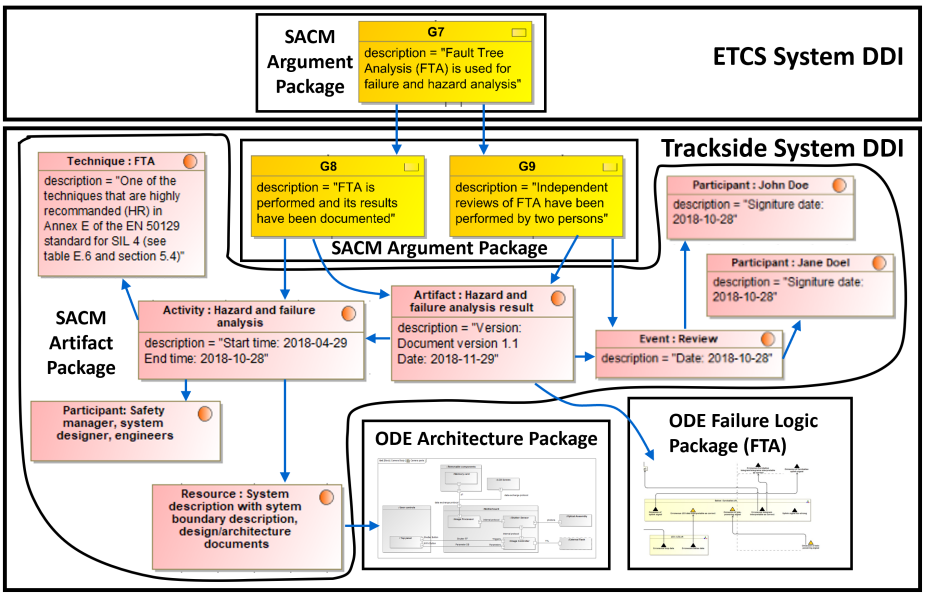

Fig. 6. ETCS FTA process evidence analysis with DDIs

cross-company safety certification. However, it only includes architectural viewpoints, safety analysis, and hazard models, missing the very important aspect of safety argumentation and evidence models, which is the backbone of a DDI. In the AMASS project, the focus rests on the organization of the safety argument and in particular the lifecycle of evidence, formalized in the *Common Assurance and Certification Metamodel* (CACM) [12]. Although this is very important in order to fulfill standard process compliance, the lack of product-related model semantics such as concrete architecture models, failure logic or hazard models disallows a proper product-related safety reasoning required for a complete compelling safety case. DDIs extend the state of the art in model-based safety certification in that they provide a dependability-specific integrated set of data models and an automation engine that together enable efficient and comprehensive safety reasoning across multiple safety aspects and supply chains.

6 Conclusion and Future Work

In this paper, we presented an approach to automatically verify safety requirements by model-based reasoning about multiple safety aspects at once that were only considered in isolation in the past. The automation is enabled by establishing formal traceability between safety argument and evidence supported by concrete product safety models all packaged together in the *Digital Dependability Identity (DDI)*. DDIs currently contain SACM safety case models integrated with hazard and risk models (HARA), architecture, failure propagation. Based

on DDIs and the corresponding DDI scripting framework, we showed that safety requirements in the railway domain can be automatically verified.

Our approach reduces the effort for creation and maintenance of a system safety case together with all relevant supporting models by first performing automated safety analysis, verification and safety artifact generation activities, second, supporting a model-based multi-tier safety engineering process and third, eliminating the human error source by relying on DDI scripts to encode safety engineering activities. The evaluation in the industrial case study suggested that the DDI approach is feasible for performing safety engineering processes faster and achieving a better process execution and safety case quality in general.

In the future, we will enrich DDIs with additional dependability aspects such as security engineering (attack trees, threat and risk analysis). In addition, we are currently performing a broader evaluation of the DDI framework in application scenarios such as autonomous driving and health applications.

Acknowledgement. The work presented in this paper was created in context of the *DEIS Project* funded by the European Commission (Grant No. 732242).

References

1. DEIS Project Consortium: Project Publications (Accessed: 30/05/2019), http://www.deis-project.eu/dissemination/
2. European Committee for Electrotechnical Standardization (CENELEC): CENELEC EN 50129: Railway application – Communications, signaling and processing systems – Safety related electronic systems for signaling (2003)
3. International Organization for Standardization (ISO): ISO 26262: Road vehicles - Functional safety (2011)
4. Kaiser, B., Liggesmeyer, P., Mäckel, O.: A new component concept for fault trees. In: Proc. of the 8th Australian Workshop on Safety Crit. Systems and Software (2003)
5. Kelly, T.P.: Systematic Approach to Safety Case Management. In: Proc. of SAE 2004 World Congress (2004)
6. Kelly, T., Weaver, R.: The goal structuring notation - a safety argument notation. In: Proc. of the dependable systems and networks workshop (2004)
7. Object Management Group: Structured Assurance Case Metamodel 2.0 (SACM) (2018), https://www.omg.org/spec/SACM/
8. Papadopoulos, Y., McDermid, J.A.: Hierarchically performed hazard origin and propagation studies. In: Computer Safety, Reliability and Security (1999)
9. Pohl, K., Hönninger, H., Achatz, R., Broy, M. (eds.): Model-Based Engineering of Embedded Systems – The SPES 2020 Methodology (2012)
10. Schneider, D., et al.: WAP: Digital Dependability Identities. In: IEEE Int. Symposium on Software Reliability Engineering (ISSRE). pp. 324–329 (2015)
11. UNISIG: ETCS/ERTMS Safety Requirements for the Technical Interoperability of ETCS in Levels (Subset-091, Issue: 3.6.0) (2015)
12. de la Vara, J.L., et al.: Model-based specification of safety compliance needs for critical systems: A holistic generic metamodel. Information and Software Technology **72**, 16–30 (2016)