Design Decisions in the Construction of Traceability Information Models for Safe Automotive Systems

Jan-Philipp Steghöfer*, Björn Koopmann†, Jan Steffen Becker†, Mikaela Törnlund*,
Yulla Ibrahim*, and Mazen Mohamad*
**Chalmers University of Technology* | *University of Gothenburg*, Gothenburg, Sweden
jan-philipp.steghofer@gu.se, tornlundmikaela@gmail.com, yulla.ibraheem@gmail.com, mazen.mohamad@gu.se
†*OFFIS e.V.*, Oldenburg, Germany
bjoern.koopmann@offis.de, jan.steffen.becker@offis.de

Abstract—**Traceability management relies on a supporting model, the traceability information model (TIM), that defines which types of relationships exist between which artifacts and contains additional constraints such as multiplicities. Constructing a TIM that is fit for purpose is crucial to ensure that a traceability strategy yields the desired benefits. However, which design decisions are critical in the construction of TIMs and which impact they have on the usefulness and applicability of traceability is still an open question. In this paper, we use two cases of TIMs constructed for safety-critical, automotive systems with industrial safety experts, to identify key design decisions. We also propose a comparison scheme for TIMs based on a systematic literature review and evaluate the two cases as well as TIMs from the literature according to the scheme. Based on our analyses, we thus derive key insights into TIM construction and the design decisions that ensure that a TIM is fit for purpose.**

Index Terms—**Traceability, Safety Assurance, ISO 26262, Safety Case, Design Decisions, Traceability Information Model**

I. Introduction

An important aspect of establishing a traceability strategy [1], i.e., a comprehensive approach to create, work with, and manage traceability links, is which concrete trace link types exist and which artifact types they connect. This information is usually recorded in a *traceability information model* or TIM [2]. While TIMs have been considered in research before, there is still a lack of understanding of how they are constructed to be fit for purpose, in particular in industrial settings [3]. TIMs in the literature are often highly specific to a certain use case (e.g., for tracing for bug localization [4]) or to a certain technique that is being used (e.g., for graph-based traceability [5]). Other works provide high-level guidelines for TIM construction (see, e.g., [6]),

What is lacking, however, is an understanding of the design space for TIMs and which impact decisions in the design of a TIM have on its applicability in a specific context. It is not clear how a TIM can be made *fit for purpose*, i.e., suitable for the task at hand within the context in which it is used. Likewise, it is not clear how to evaluate TIMs from the literature when selecting a published TIM as a foundation for use in a development team or organisation. To address these issues, this paper answers the following research questions:

RQ1 Which critical design decisions are made in the development of traceability information models?

RQ2 What are important drivers of design decisions in the development of traceability information models?

RQ3 How do different traceability information models compare to each other in terms of relevant categories?

Methodologically, we report on two case studies that each comprised the construction of a TIM. Each case study was performed by a distinct subset of the authors of this paper, in collaboration with industrial partners and domain experts. Both TIMs are from the automotive domain and are intended for use in projects that are safety-critical. They were constructed based on different premises and for different purposes: one as a general, abstract TIM for all sorts of end user functions and one with a specific system in mind. We analyze both cases for critical design decisions and discuss the impact of these decisions. We also characterize both TIMs as well as additional ones from the literature using a set of criteria derived from a lightweight systematic literature review. This review allowed us to identify criteria that help understand the consequences of design decisions for different TIMs and provides insights into which choices can be made when designing TIMs.

The contribution of this paper is thus three-fold: 1) we present two cases where TIMs were constructed for similar, but slightly different purposes with industry participation and describe critical design decisions along with their drivers; 2) we define a comparison and evaluation schema for TIMs based on important design decisions or their outcomes; and 3) we compare the two TIMs from our cases with two TIMs from the literature and show important differences and commonalities. Overall, this paper contributes to the body of knowledge on how to structure traceability information, how to construct TIMs, and systematizes the existing work.

II. Background and Related Work

In this section, we first discuss the topic of traceability for automotive safety. We then introduce the concept of TIMs before discussing support for the construction of TIMs in the literature and finally introduce some concrete example models that have been suggested in related work.

A. *Traceability for Automotive Safety*

Safety standards, such as ISO 26262 [7] in the automotive domain, specify various traceability requirements. ISO 26262 does not explicitly define the types of trace links or the structure of a TIM. However, it defines:

1) the types of artifacts that are created during the development process;
2) the information about artifact relationships that needs to be maintained; as well as
3) requirements for documentation.

For example, ISO 26262 requires that requirements shall be traceable to requirements on upper hierarchical levels (respectively to the identified hazards at the top level), the hardware and software elements implementing them, and the specification and result of verification activities related to them [7, part 8]. Note that ISO 26262 does not specify how traceability information shall be stored. Hence, it is both valid to store trace links as part of the artifact itself or, for example, as a separate database or even a mixed form of that. Also, it is up to the traceability tool to store only parts of the information (e.g., a single direction of a link) as long as all other required traceability information can be derived.

During product development, artifacts evolve over time. The *hardware-software interface*, e.g., is first defined during system development and later refined in the software and hardware development phases. Changes later on in the development process must be traced back to earlier phases, because they may introduce new requirements or points of failure. Furthermore, changes can lead to outdated verification results and thus make repeating verification & validation (V&V) steps necessary. To avoid inconsistencies between artifacts due to independently evolving artifacts and different product configurations, ISO 26262 requires that all artifacts undergo version and configuration management [7, part 8]. In that regard, traceability management is both an opportunity and a challenge. A traceability management tool and a well-defined TIM are key enablers for a change impact analysis as required by safety standards [8]. On the other hand, traceability management needs to handle different document versions.

Besides assisting in change impact analysis, traceability is also important for an overall argument about product safety and compliance with the safety standard. This argument is referred to as *safety case* in ISO 26262. Providing the artifact documentation alone is not sufficient evidence for safety – the argument *how* the documented artifacts and activities support claims about the product's functional safety is also required. Traceability – and, again, a well-defined TIM – are key here [9], as they provide, e.g., the relations between requirements and implementation, and connect them with the V&V plans and results that certify their completeness and correctness.

B. Traceability Information Models

Following the definitions by Maro et al. [3], Mäder et al. [10], Rempel and Mäder [2], as well as Ramesh and Jarke [11] and using the terminology from Gotel et al. [12], a TIM defines two important aspects:

Trace Artifact Type Describes a specific kind of element that can be connected by one or more trace link types. Examples include requirements, C functions, test cases, or successful builds. The instantiation of a trace artifact type is a *trace artifact* – a single "traceable unit of data" [12] such as a specific requirement, C function, or test case.

Trace Link Type Defines which relationships between which artifact types are permissible and imposes constraints such as multiplicities. It can carry additional information such as a name which often also defines the semantics or meaning of the link type (e.g., "satisfies" or "implements") or meta-data such as creation date or creating user. Its instantiation is a *trace link*, the concrete relationship between two specific trace artifacts.

The literature commonly depicts TIMs as UML class diagrams where trace artifact types are usually represented as classes, and trace link types are often represented as associations between the trace artifact types (see, e.g., [6], [13]–[15]). Alternatively, the trace link type can also be represented as a class and which trace artifact types it connects is shown by associations (see, e.g., [16], [17]). In case a trace link type needs to carry additional meta-information this can be expressed as an association class (as in, e.g., [8]). From a model-driven engineering perspective, the TIM is defined as a meta-model on level M2 [18] and is therefore also sometimes called "traceability meta-model" (e.g., by Aizenbud et al. [19]). Most literature, however, uses this term to denote an even more abstract view (see, e.g., [20]), usually consisting of meta-classes for traceable artifact types and trace link types. An older, but still relevant term is "traceability scheme" (see, e.g., [21], [22]) which often encompasses more than a TIM, e.g., the rights of different stakeholders to modify trace links.

The concrete instantiation of a TIM can take different forms: Trace links can be stored within artifacts (e.g., the Javadoc of a test method can refer to the requirement that is being tested) or stored in databases and even spreadsheets. Model-driven approaches use specialized trace models [23]. However, the TIM still defines which trace links are permissible as well as the format of the trace links [19].

Formalized TIMs also enable automated reasoning about traceability [11] and the automated creation of trace links, e.g., via model transformations (see, e.g., [24]). Apart from these technical aspects, TIMs also help to document and standardize traceability efforts in an organization [6], in particular if there is accompanying informal information that captures rationale and background [11].

C. Literature Support for the Construction of TIMs

While there are a number of works that propose concrete TIMs (see Section II-D), only few papers provide guidelines for the construction of new TIMs from scratch.

Mäder et al. [10] provide a few guidelines for defining TIMs and for traceability in safety-critical environments. They also discuss some guidelines on TIM design [6]. However, these guidelines are on a rather generic level, e.g., that a clear definition of the trace granularity is required in the TIM. A similar level of detail is achieved by Cleland-Huang et al. [25] who also emphasize fitness for purpose and suitable granularity. Lago et al. [26] suggest a "scoped" approach that uses typical developer activities as a starting point to devise "traceability paths". These are not formalized into TIMs in the sense of Section II-B, though.

The work by Ramesh and Jarke [11] provides some reference TIMs that practitioners could use as a starting point for designing their own TIMs. Depending on ambition and needs, they propose low-end and high-end TIMs. Importantly, the authors also define an abstract conceptual model of traceability that illustrates which questions need to be asked when constructing TIMs. In particular, they suggest to ask specific questions about artifact and link types to construct a TIM: 1) *What* information is represented? 2) *Who* are the stakeholders? 3) *Where* is the information stored? *How* is the information represented? 5) *Why* was information created or modified? and 6) *When* was the information captured or modified?

Maro et al. [3] make the point that the current guidelines are not addressing the challenges of practitioners when constructing TIMs. They point out that product complexity, company size and distribution, as well as long-lasting products impose a number of issues that are not addressed at the moment. Product lines, the need to evolve TIMs over time and their role as spanning teams, organizations, and development processes are still open research issues.

D. Concrete TIMs for Safety-critical Systems

Katta et al. [27] differentiate between two types of traceability links: those related to "normal" development work and those related to safety assessment. This distinction, based on previous work [28], is represented in the TIM which contains separate areas for these types. The TIM is also specified on different "levels of abstraction", that roughly follow the refinement of a system from concept to implementation. One of the main purposes is to identify the impact of a requirement change on the safety case. This capability is illustrated by the example of an air traffic control system.

Taguchi et al. [29] present a meta-model for TIMs for the construction of safety cases. It connects `solution` and `context` from the *Goal Structuring Notation* (GSN) [30] to the `traceable artifact` and `traceable unit`. Classes for the `safety lifecycle` are split into development `phases`. This means that the instantiations of the TIM are specific to artifacts relevant in a development phase. Links can also connect artifacts across development phases. The TIM is instantiated for a case in the railway domain for three early phases of the design process.

Nair et al. [8] present SafeTIM, a general purpose TIM for projects that require safety assessment, in particular for tracing to safety evidence. The authors list the different purposes traceability links can have in this use case, in particular to provide a "chain of evidence". Arguably, what the authors are presenting is not a true TIM since many of the concepts in SafeTIM are not traceable artifacts, but rather properties of other concepts (e.g., the `ArtifactRelationship` concept) or managerial concepts without explicit representation as artifacts (e.g., the `Project`).

Agrawal et al. [31] describe the automated generation of safety cases based on trace links. The underlying TIM connects relevant artifact types. Trace links based on this TIM can then be traversed to create an *artifact tree* which, in turn, is transformed to a safety tree. The TIM used by the authors does not contain meta-information such as versions for the different artifacts, but trace links types with specific semantics (e.g., refinement and implementation).

III. TIM I: For Use at an Automotive OEM

The first analyzed TIM has been developed for an engineering team at an automotive OEM. This team manages safety cases for the end-user functions they provide based on ISO 26262 [7] and struggled with maintaining these safety cases when the underlying artifacts were updated. The industrial partner stated these goals for the introduction of a TIM:

1) Simplify the maintenance of safety cases from the safety engineer's point of view.
2) Improve the ability to track changes in the artifacts with an impact on the safety case from the developer's point of view.
3) Improve the ability to use artifacts from a variety of sources in the construction and maintenance of a safety case from the safety engineer's point of view.

The industrial partner approached the researchers with the request to address these goals by helping them to introduce trace links to artifacts relevant in the argument or as evidence. Together, they identified that the first step towards this overarching goal will be to construct a suitable TIM.

A. Methodology for Creation of the TIM

The TIM was developed in three iterations in collaboration between researchers and the industrial partner, roughly following a design science research approach [32]. Each iteration consisted of a *problem awareness phase* in which the problem at hand as well as the results of the previous iteration were analyzed, a *suggestion and development phase* in which the TIM was further developed, as well as an *evaluation phase* in which the results were presented to the industrial partners in order to gather feedback to start the next iteration.

Iteration 1: The foundation for the construction of the TIM were the TIMs proposed by Nair et al. [8] and Taguchi et al. [29]. A preliminary evaluation showed that these existing TIMs were not directly applicable since they were not adapted to the automotive domain and ISO 26262. Additional material were guidelines on how to create safety cases as well as a complete safety case provided by the industrial partner. Along with the ISO 26262 standard, these documents showed which artifacts existed and needed to be traced. Furthermore, the researchers participated in meetings of the industrial partner in which the safety case and its construction were discussed. These observations provided further input into the artifacts that play a role in the safety case and how they relate.

The draft TIM was reviewed in a focus group with several engineers from the industrial partner. Feedback was mainly provided on the level of granularity of the TIM. The engineers

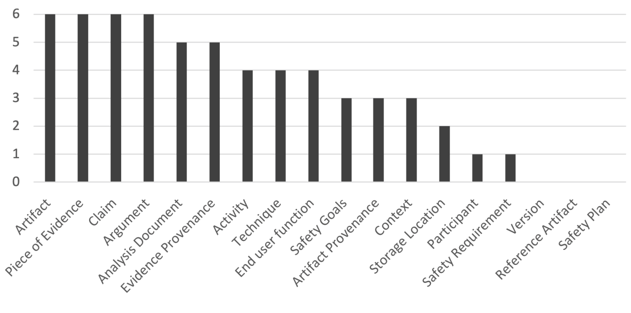

Fig. 1. The number of evaluated safety cases which mentioned a certain concept. A total of six safety cases were evaluated.

remarked that splitting evidence into acceptance tests, code, tests, simulations, etc., following the approach by Agrawal et al. [31] will create a trace model that is too complex and large to be used in industrial processes.

Iteration 2: The second iteration was based on the feedback from iteration 1 as well as a deeper analysis of how the industrial partner works with safety plans. A key insight was the need to represent the hierarchy of safety requirements as well as their relation to evidence and claims [33]. The revised version of the TIM included changes mainly related to the level of granularity as well as the linked artifacts. It also introduced additional meta-data, e.g., storage locations that are important to retrieve the artifacts when needed.

The updated TIM was again reviewed as part of a focus group. Even though some artifacts had been removed, the TIM was still deemed as too granular. A distinction between different analysis documents, for example, was not seen as advantageous. The artifacts `Acceptance Process`, `Acceptance Plan`, and `Acceptance Criteria` suggested by Taguchi et al. [29] were deemed out of scope.

Iteration 3: In the final iteration, the feedback from the experts was augmented with an analysis of five safety cases from literature [34]–[38]. These cases have been selected based on their size and complexity, their completeness, whether they include relationships to evidence, how credible the source was, and their relevance to the automotive domain. There were two main analyses performed using the safety cases:

1) The researchers performed a frequency analysis to determine which concepts were referenced in the company's safety case and the safety cases from the literature (cf. Figure 1). This provided insight into common concepts in these cases which need to be traced.
2) The researchers also mapped the draft TIM to the three safety cases that provided the best coverage of artifacts, including the company's safety case (as well as [35] and [34]). This provided insights into the relationships between the artifacts and if they are captured by the draft TIM. Using the company's own case also made it easier to communicate the results to the practitioners.

Based on the findings of the analyses, an updated version of the TIM was constructed and, again, evaluated in a focus group with four practitioners. Topics of discussion were *applicability in an industrial setting*, the *challenges practitioners foresaw*, the *understandability and accuracy of the model*, how the TIM could be augmented to *facilitate adoption*, and the *level of support for creating the necessary trace links*. Another aspect that was addressed is the extent of compliance with ISO 26262 in order to meet all obligations. These discussions led to a final revision of the TIM.

B. Key Elements of the TIM

The final TIM is shown in Figure 2. It follows the distinction in a development and a safety assessment sphere proposed in [28]. The part related to the safety case describes the relations among artifacts that construct and specify the functional safety case. It also documents references to the development process elements that are required for maintaining a safety case. Some key elements of the TIM are:

Artifact Since the TIM will be used in the context of ISO 26262, an artifact represents a work product. The meta-class is further refined into `Result`, `Implementation`, `Design`, and `Plan` artifacts, concepts not depicted in the figure.

Storage Location Since artifacts are usually stored in different repositories or systems, the storage location is explicitly recorded to enable retrieval of an artifact. In practice, this meta-class is not a separate entity, but rather meta-data of the artifact.

Participant The `Participant` identifies the responsible engineer for a certain artifact. This is important since 1) it allows others to identify who to speak to in case of questions about the artifact; 2) it establishes accountability; and 3) part of the safety argument is that the engineers involved in the creation of a safety-critical system have the credentials necessary for this kind of work which requires identifying them.

Scope The `Scope` meta-class was added to the TIM as a result of the analysis of safety cases. It captures a number of pieces of information not associated with specific artifacts directly, such as assumptions about the system, the context and environment in which it is executed, or the relevant *Automotive Safety Integrity Level* (ASIL) that determines the integrity requirements for the function.

Activity Including the `Activity` that creates an artifact allows to show that all activities that are demanded by a safety standard are performed. Additionally, a link between activity and artifact indicates that when it is performed, an artifact is updated and, consequently, the safety case might be updated.

C. Critical Design Decisions and Their Drivers

In the following, we list the key design decisions when creating the TIM along with the drivers behind these decisions.

Coverage of Relevant Artifacts: Notably, later iterations of TIM I removed the requirements. Initial versions contained safety requirements, acceptance criteria, and safety goals, but all of these concepts were removed since the industrial partner did not deem them necessary and only one of six

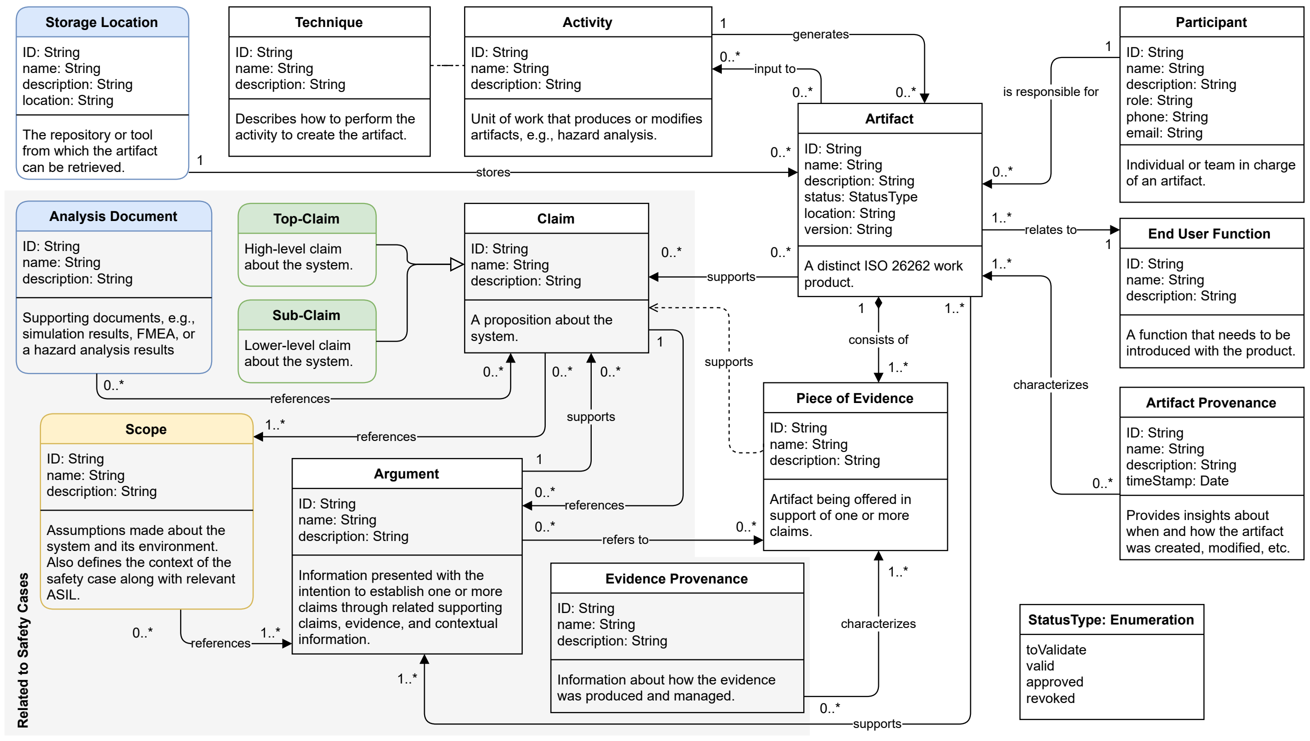

Fig. 2. A simplified version of the TIM created for the OEM. Blue, yellow, and green indicate that the concept was added in Iteration 2, 3, and 4, respectively.

safety cases that were used for validation related to them (cf. Figure 1). Requirements are instead now implicitly represented by `Claim`. This might seem surprising at first, but when considering the purpose of the TIM, i.e., to support the construction and maintenance of safety cases, this omission becomes more understandable: the scope of the TIM is limited to what ends up in the safety case and since requirements are represented only through claims, they were removed from the TIM. *Fitness for purpose* as well as *minimal traceability information* were thus drivers of this decision.

Concreteness of Artifact Types: While the artifact is refined into more concrete sub-types, the refinement does not specify which concrete artifact types are being used, e.g., to represent an analysis result. This was a conscious choice since this makes the TIM independent from a concrete development effort and allows different teams to refine it for their own development environment. For the same reason, the inner structure of the artifacts (e.g., what concrete elements are contained in a design artifact or that requirements can be hierarchical) are ignored. On the other hand, this also leaves room for interpretation and ambiguity that counteracts the purpose of a TIM to standardize the traceability effort within a development team or an organization. The drivers behind these decisions were thus the *genericity* of the TIM and its inclusion in existing processes.

Recording Rationale and Storage Location: The `Artifact Provenance` and `Evidence Provenance` meta-classes make the need to record the rationale of changes to artifacts and evidence explicit. Together, this information can form an audit log that, in connection with the version of an artifact, provides insights into how the development artifacts and their use in the safety case have evolved. Together with the explicit `Storage Location` this allows retrieval of all relevant artifacts and reconstruction of any changes that occurred. This ability is modeled in a similar way in SafeTIM [8]. The driver behind these decisions was thus *adherence to a safety standard*.

IV. TIM II: Driver-Assistance System

The second TIM has been created to allow traceability between artifacts involved in the development of a driver-assistance system. The description of the system has originally been part of a challenge at the *10th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems* (WATERS) [39] and was formulated by engineers from Robert Bosch GmbH and scientists from the University of Modena and Reggio Emilia. The original artifact set consisted of a textual description in PDF format as well as an AMALTHEA model for Eclipse APP4MC[1] that describes relevant aspects of the hardware and software. The system in focus is intended to calculate proper throttle, steering, and brake signals to follow a map of predetermined waypoints. In addition, a high-priority task to break and perform emergency maneuvers to avoid obstacles is included.

[1]https://www.eclipse.org/app4mc/

Based on the assumptions of an operation in urban areas and the implementation of level 3 automation, the initial description was greatly extended by a team of experts and researchers to create the MobSTr dataset [40]. In a series of workshops, they developed a list of additional requirements and architectural designs that served as a basis for a *Hazard Analysis and Risk Assessment* (HARA) according to ISO 26262. The identified hazards were then used to derive a number of exemplary safety goals including sets of subordinate safety requirements. Moreover, different safety analyses such as a *Failure Mode and Effects Analysis* (FMEA) and a *Fault Tree Analysis* (FTA) have been performed to refine the initial requirements. Finally, an overarching safety case has been defined. Most of the artifacts have been validated by experts from an automotive tier-1 supplier before proceeding.

A. Methodology for Creation of the TIM

The TIM for the driver-assistance system was developed in an iterative process based on a series of five workshops and regular, biweekly meetings. Four safety experts (one from industry, three from academia) and one traceability expert from academia were involved in the preparation of the additional design artifacts and the creation of the TIM in focus.

As a preparatory activity, the artifacts listed above were created and validated by the involved experts. Before starting the construction of the TIM, four main traceability goals were then agreed upon:

1) Improve the adherence to common safety standards from the safety engineer's perspective.
2) Simplify the construction of a safety argument from the safety engineer's perspective.
3) Increase the efficiency of identifying artifacts related to an analysis result from the developer's point of view.
4) Increase the efficiency of showing that all requirements are fulfilled from the developer's point of view.

Based on these goals and the requirements presented in Section II-A, a general information model containing all relevant artifacts along with their document types was derived. Besides a number of Excel spreadsheets, Eclipse Papyrus[2] models, and *Open Dependability Exchange* (ODE) [41] models, the original system description in PDF as well as the AMALTHEA model provided as part of the challenge were included. The information model was then refined in several iterations and tailored to be fit for purpose through appropriate detailing.

Finally, the TIM was instantiated from the general information model by concretizing the multiplicities of the artifact relations. In order to facilitate the practical use, the TIM was implemented in the traceability management tool Eclipse Capra [42]. Repeated reviews by the safety experts ensured the validity of the results.

B. Key Elements of the TIM

The TIM created to support the development of the driver-assistance system is presented in Figure 3. It consists of 22 meta-classes, which have been partially combined into a `Requirement Specification` package, different safety analyses (such as FMEA and FTA), as well as an `AMALTHEA Model` package. Some key elements of the TIM are:

[2]https://www.eclipse.org/papyrus/

Item Definition The `Item Definition` contains the challenge description including a definition of the system's functionality, a list of software tasks as well as assumptions about the technical realization. The explanations are supplemented by `Additional Assumptions` developed during the extension.

Requirement Based on these two artifacts, different kinds of `Requirements` were derived. The top-level safety requirements, which are referred to as `Safety Goals`, address the `Hazards` identified as part of the HARA. Each safety goal is refined into a set of lower-level `Safety Requirements` that specify the required `Measures` developed as part of the safety analyses to prevent the occurrence of `Hazards`.

Component The system itself is modeled by a set of `Components`. They realize the `Item Definition` and are each linked to a number of `Requirements`, which are refined in parallel with the component's decomposition. In our example, the `Components` are implemented by `Runnables` of the `AMALTHEA Model`.

Safety Analyses According to ISO 26262, each `Component` has a number of `Failure Modes`, the `Effects` of which can lead to the occurrence of `Hazards`. These relations are typically investigated in an FMEA, to which a separate package has been dedicated here. In contrast, the complementary `Fault Tree Analysis` that details potential causes of the `Hazards` was modeled as a single meta-class based on the work status of the artifacts.

Safety Case The `Safety Case` is the heart of the TIM and links all presented artifacts – either directly or indirectly – into an overarching safety argumentation. It references the identification and mitigation of all relevant `Hazards` as *goals*, describes *claims* in terms of `Safety Requirements`, and provides *evidence* from `Failure Modes and Effects Analysis`, `Fault Tree Analysis`, and additional `V&V Analysis` to substantiate the safety argument.

C. Critical Design Decisions and Their Drivers

Again, we list the most crucial design decisions for the TIM creation along with the drivers behind those decisions. As in the case of TIM I, the *adherence to a safety standard*, i.e., to ISO 26262, was one of the most important high-level drivers and thus had an influence on almost all design decisions.

Internal Trace Links: Some of the trace links are handled within the artifacts, e.g., the allocation of `Requirements` to `Components` or the refinement of `Requirements` by others. This fact results from the templates and meta-models used for the artifact creation, which already map some of the relationships completely. Since this form of linking makes the dependencies already visible during development, the explicit instantiation of trace links could be omitted. The main drivers

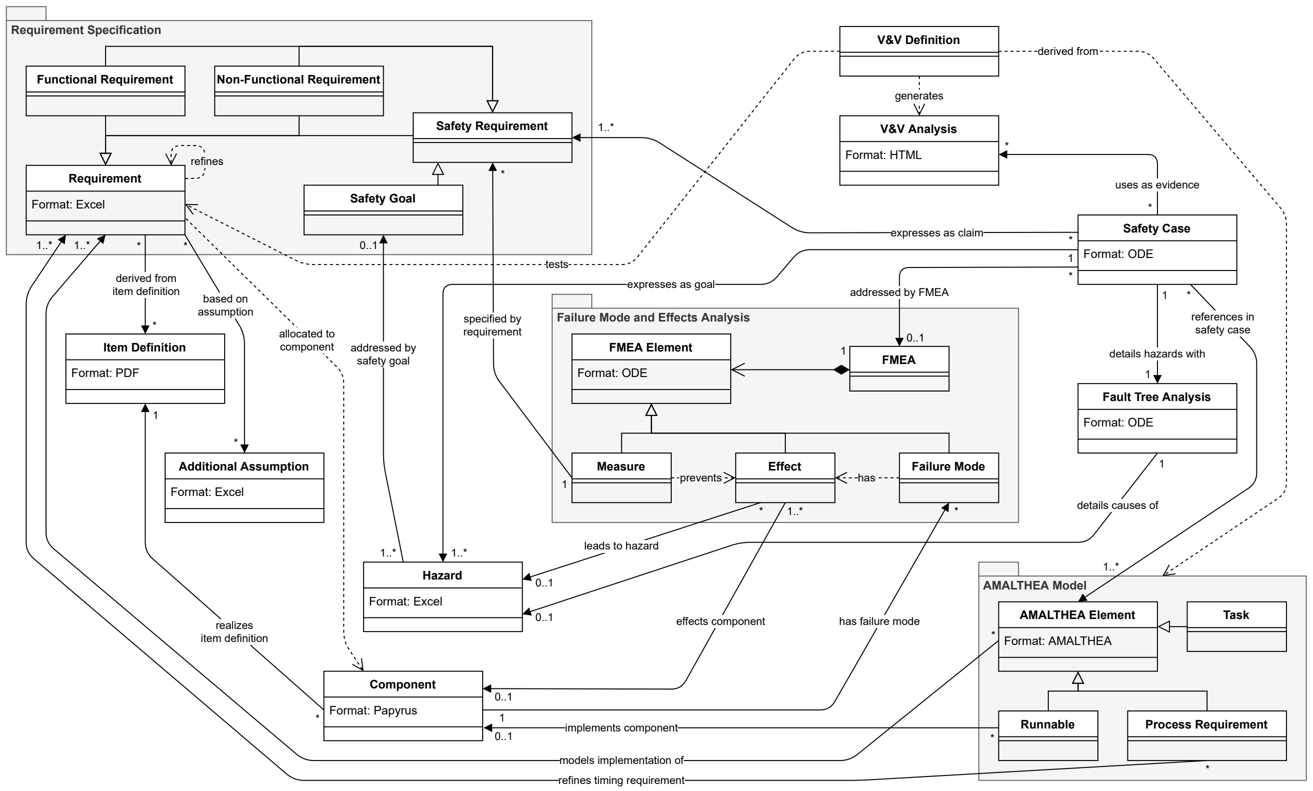

Fig. 3. The traceability information model created to support the development of the driver-assistance system. Dashed lines represent relationships that are not captured in the TIM since they are either implicit or captured in the DSLs themselves.

behind this decision were thus the *light-weight documentation of traceability information* as well as the *fitness for purpose* as it simplifies reviews and discussions.

Reverse Tracing: In contrast to the usual life cycle, tracing in TIM II does not originate from the requirements, but the other way around. This reflects the way the system was developed. Since the AMALTHEA model published for the original challenge detailed the characteristics of the hardware and software, but omitted artifacts important for the safety assessment in early design phases, the corresponding artifacts had to be added afterwards. Therefore, the team of experts reverse engineered relevant safety artifacts based on the high-level challenge description and created the trace links. The reverse tracing decision results from the *parallel evolution of artifacts and traceability information* and is promoted by the step-by-step approach of the iterative development process.

Level of Granularity: Since TIM II was developed with a specific system in mind and refined during its evolution, the level of granularity is based on the work status of the artifacts considered. One example of this is the varying granularity in modeling safety analysis results. In contrast to the FMEA, which was carried out relatively early in the creation process, the FTA was still in the process of being created at the time of submission. For this reason, the FTA is presented in an abstract way and is not (yet) broken down into further meta-classes. The drivers of these decisions were *fitness for purpose*, *minimal traceability information*, and the *ongoing development*. In the future, we expect the TIM to evolve further as the development of the case study continues.

Technical Realization: The design of TIM II was additionally influenced by technical aspects. For example, the names of the meta-classes and trace links had to be unique. As an advantage, type safety can be ensured by the tooling which facilitates both automation and consistent error handling. *Applicability* as well as the *purpose*, i.e., intended use in timing/safety analyses were thus drivers for these decisions.

V. Comparison of the TIMs

Based on the description of the two TIMs, we will first compare TIM I and TIM II in terms of the design decisions that had a different outcome and what was driving these decisions. We then introduce a set of criteria that allows comparing TIMs based on the outcomes of the design decisions and how different TIMs from this paper and the literature compare to each other. In the following, we will first consider some design decisions and their drivers before discussing the high-level drivers for both TIMs. These parts thus answer **RQ1** and **RQ2**.

A. Crucial Differences in Design Decisions and Drivers

When reviewing the critical design decisions, we observed a number of key differences between the two TIMs. These differences are spelled out in Table I.

1) Some Concrete Decisions and Their Drivers: The decision about coverage of relevant artifacts and which artifact types to include was a major point of discussion for TIM I, but was decided very quickly for TIM II. Drivers behind the decisions which artifacts to include were the *purpose* of traceability as well as *minimal traceability information* in order to reduce overhead in both cases.

Deciding about concreteness of artifact types and its related topic granularity was relevant for both TIMs. For TIM I, the choice to remain on a high level of abstraction and not make artifact types concrete was driven by *genericity* – more concrete types would have impeded the ability to apply the TIM in different projects. Specifying the granularity on which is traced was not considered relevant. For TIM II, on the other hand, how concrete the artifact types were as well as the granularity was driven by the *purpose* – the ability to trace all relevant safety assurance artifacts. This meant to be very concrete about artifact types, but also adjust the granularity level to ensure that *minimal traceability information* is collected.

This is visible since TIM II includes fine-grained types for some safety analysis aspects (e.g., a breakdown of `FMEA` elements into `Failure Mode`, `Effect`, and `Measure`), but considers the safety case as a "monolithic block" without detailed structuring into more fine-grained elements. Which role certain artifact types play in the safety case is denoted via the trace link types (e.g., `uses as evidence`). In contrast, TIM I contains `Claim`, `Argument`, and `Piece of Evidence`. This means that TIM I is *role-focused*, i.e., focused on the role an artifact plays in safety analysis, whereas TIM II is *type-focused*, i.e., focused on which types are relevant in safety analysis.

Furthermore, TIM I contains `Activity` as an explicit concept. Trace links that connect artifacts and the activities that created them, are represented, e.g., in a process model. In TIM II, however, activities are mentioned implicitly via the artifacts they create. This implies that TIM I is *process-driven* whereas TIM II is *work product-driven*. This is also witnessed by the inclusion of the `Participant` in TIM I and the focus on provenance for both artifacts and evidence.

2) High-Level Drivers: Besides these concrete drivers, the development of the two TIMs was significantly influenced by a number of high-level drivers. For TIM I, the *applicability in an industrial context*, the *challenges practitioners foresaw*, and the *understandability and accuracy of the model* played a key role in all design decisions. In addition, possible changes to *facilitate adoption* and to conform with the existing *level of support for creating the necessary trace links* were considered. In contrast, the design of TIM II was primarily influenced by its development in an *academic context* supported by industrial experts. Besides the primary focus on the *adherence to a safety standard*, the TIM creation was driven by *frequent incremental updates* as well as intended enhancements in an *ongoing development effort*.

TABLE I
CRITICAL DESIGN DECISIONS AND THEIR DRIVERS FOR THE TWO TIMS.

Decision	TIM	Driver
Coverage of artifacts	TIM I	*Purpose*, *minimal traceability information*
Concreteness of artifact types		*Genericity*
Rationale and storage		*Adherence to a safety standard*
Design approach		*Process-driven*
Artifact focus		*Role-focused*
Coverage of artifacts	TIM II	*Purpose*, *adherence to a safety standard*
Internal trace links		*Light-weight documentation of traceability information*, *purpose*
Trace direction		*Parallel evolution of artifacts and traceability information*
Granularity		*Purpose*, *minimal traceability information*, *ongoing development*
Tooling		*Applicability*, *purpose*
Design approach		*Work product-driven*
Artifact focus		*Type-focused*

B. Criteria for the Comparison of TIMs

In order to define important criteria for the comparison of TIMs, we screened the available literature using a lightweight systematic literature review approach. We analyzed the first 150 hits found by Google Scholar for the search term "traceability information model". We decided that our results are saturated by determining that we have covered a small set of known good results [3], [8], [11], [28], [29] and that at least 20 included hits did not add new criteria. We excluded non-English documents, documents that are not about traceability in software engineering (but rather, e.g., in manufacturing or supply chain management), documents whose full-text was not available, and non-peer reviewed documents (e.g., books).

After applying these exclusion criteria, we screened the remaining 98 papers and disregarded those that did not discuss criteria for the design of TIMs. Snowballing added two additional papers. We then analyzed the selected 39 papers and extracted relevant criteria for comparison of TIMs. Such criteria were usually either mentioned as design goals for TIMs or as evaluation criteria. The candidates were collected and sorted in a spreadsheet before we merged them and derived definitions for each of them from the available sources.

Notably, even the literature that includes TIMs rarely discusses the design decisions and the considered alternatives, but rather only presents the final outcome. However, the criteria we identified still reflect design decisions: the level of granularity or whether the TIM is strongly typed, e.g., are characteristics based on design choices. While the criteria are thus useful for comparing TIMs, they also reflect important design decisions. Table II shows the criteria and their evaluation for four different TIMs. The table also contains the answer for **RQ3**.

A number of papers mentioned link semantics (e.g., [27], [46], [51], [52]), but none of them made it explicit what

TABLE II
CRITERIA FOR THE EVALUATION OF TIMS AND THEIR VALUES FOR FOUR DIFFERENT TIMS. POSSIBLE VALUES ARE HIGHLIGHTED IN ITALICS. THE TABLE IS STRUCTURED INTO ROUGH CATEGORIES FOR THE CRITERIA: PURPOSE, APPROACH, STRUCTURE, AND REDUNDANCY.

Criterion	**Defining Questions and *Possible Values***	**Selected Sources**	**TIM I**	**TIM II**	**SafeTIM [8]**	**TIM for RAMS [29]**
Stated Purpose	Is the purpose of the TIM clearly stated (*defined*)? Are the different stakeholders and their respective needs identified in that purpose (*fully defined*)?	[8], [12], [14], [24], [43], [44]	defined	defined	defined	defined
Coverage	Does the TIM provide *partial* or *full* coverage of the artifacts required to fulfill its purpose?	[7], [28], [29]	full	partial[1]	full	full
Specificity	Is the TIM *general purpose*, *specific to a purpose*, or even *highly specific* to a certain team, organization, or system?	[24], [45]	specific to purpose	highly specific	specific to purpose	specific to purpose
Design Approach	Is the starting point of TIM design the *process* or the *work products*?		process	work product	process	mixed[2]
Artifact Focus	Are the *roles* of the artifacts or their *type* reflected in the TIM?		role	type	role	role
Mapping	Does the TIM map to the work products in a *direct* way or is an *indirect* mapping necessary, e.g., because not all concepts in the TIMs map to artifact types and it is not unambiguous which elements of the TIM represent trace link types?		indirect	direct	indirect	direct
Typing	Are the traceable artifact types identified by generic types (*weak*)? Or are traceable artifact types more concretely identified via the meta-classes of the respective domain-specific languages (*strong*)? Are these levels *mixed*?	[17], [21], [28], [46]	weak	mixed	weak	mixed[3]
Granularity	Are trace link types defined between specific artifact types (*low-level*, e.g., between a `Non-functional Requirement` and a `Timing Constraint`) or between rather generic artifact types (*high-level*, e.g., between `Artifact` and `Claim`)? Are these levels *mixed* in the same TIM?	[6], [25], [27], [43], [46]	high-level	mixed	high-level	low-level
Constraints	Does the TIM define constraints such as *multiplicities* for the trace link types or the primary reading *directions* of trace links? Does it include additional *correctness* constraints, e.g., as OCL constraints?	[10], [12], [17], [21], [45], [47], [48]	multiplicities, directions	multiplicities, directions	multiplicities, directions	none[4]
Arity	What is the arity, i.e., how many different artifact types a certain trace link type connects (*binary*, *n-ary*)?	[5], [21]	binary	binary	ternary[5]	binary
Meta-Information	Is meta-information attached to *artifacts* (e.g., storage location or version) or to *trace links* (e.g., tracer or creation date)?	[12], [14]	artifact	none	artifact, trace link	none
Redundant Paths	Does the TIM allow a transitive connection between two artifacts on two different paths (*yes*/*no*)?	[2], [6], [49], [50]	no	no	no	yes

[1] no implementation artifacts, no test cases, etc. [2] work products are explicit, but development phases are used to structure the TIM
[3] uses elements from GSN in the TIM [4] multiplicities and reading directions only in meta-model [5] association classes for some link types

semantic information concretely should be included. From the TIMs, it could be derived that semantics mostly refer to named links that provide their meaning and sometimes multiplicities and other constraints. However, since no TIM in the literature used unnamed links (with the exception of [53] which explicitly mentions that the links carry semantics; the fact that the names are missing in the depicted TIM could thus also just be an oversight), we have not added the naming as a separate criterion.

VI. Discussion and Lessons Learned

Our systematic literature review yielded a total of 39 TIMs in 98 included studies, but the papers usually do not describe the design decisions that led to them. In addition, we found 20 traceability meta-models, i.e., higher-level models that define the concepts relevant in a TIM such as trace link type and artifact type. Most papers refer to the meta-model introduced by Ramesh and Jarke [11] for requirements traceability.

A. Fitness for Purpose and TIM Evolution

The most cited criterion for the quality of a TIM in our review was its *purpose*. In a way, the design decisions and drivers we identified in Table I and the criteria we identified in Table II all contribute to the main evaluation criterion: *fitness for purpose*. However, fitness for purpose can only be partially evaluated on a TIM: the ultimate test is using the TIM to create trace links, maintain them, and use them for the activities they were designed for. Purpose is defined for the overall traceability strategy which in turn consists of purpose, process, structure, and tools, where the TIM provides the structure [1]. Of course, these four elements are inter-dependent and the TIM needs to be aligned with the other aspects.

TIM II has been developed with this in mind and is intended to continue evolving. The FTA, e.g., was not fully integrated into the TIM at the time of writing. We anticipate that the corresponding artifacts will be added once the FTA is conducted to integrate it with the rest of the artifacts.

As with all design processes, it would be unusual if a TIM was perfect right away. This has also been acknowledged in the *Grand Challenges of Traceability* [52]. However, we only found one paper in our literature review that explicitly discusses evolving the TIM [54].

An additional, implicit purpose of a TIM should not be neglected: a TIM has *explanatory power*. The discussions with the industrial partner about TIM I showed that an overview of the concepts involved in safety case construction and how they relate helps practitioners gain insights about their work.

B. Approaching TIM Design

Since the literature does rarely provide rationales for the design of the TIM, we must mostly rely on our two case studies to discuss how TIM design is approached. In both cases, the TIM was not developed from scratch, but based on a template: for TIM I, it was SafeTIM [8] and the TIM by Taguchi et al. [29]; for TIM II, it was ISO 26262 [7] and the different high-level relationships prescribed therein (cf. Section II-A). Such a template provides a good starting point for customization. However, such reuse also implies some design decisions: since SafeTIM is *process-driven*, TIM I also ended up *process-driven*; on the other hand, since ISO 26262 is mostly concerned about work products, TIM II ended up *work product-driven*. The same is true for *role focus* and *type focus*.

Regardless of the design approach, a TIM always reflects a process. Ramesh and Jarke [11], e.g., in explaining their low-level TIM describe the requirements refinement process. The concepts, the link directionality, and other crucial elements of TIMs are often defined based on the lifecycle of information. It is therefore not unusual to see that links originate in early artifacts (e.g., requirements) and target later artifacts (e.g., design models or test cases). TIM II is unusual in the sense that the ultimate source of the trace links is the safety case. This can be attributed to *fitness for purpose* again: TIM II was designed to facilitate safety assessment, not other activities such as change impact analysis.

To a certain degree, both TIMs are conceptual models of safety cases. They include specific artifacts that play an important role in a safety case and have been constructed with safety cases in mind. However, the *specificity* differs: while TIM I has been designed to be applicable to different systems and processes, TIM II has been designed for a specific system class and a specific set of artifacts defined in specific domain-specific languages. Such a specificity is also visible in the trace links. The knowledge of the experts involved in the design of the TIM had detailed knowledge not only about the involved DSLs, but also about how they are applied. Their discussions included, e.g., details such as which model elements are modeled first. A result of such a discussion was to relate `Components` to `Runnables` instead of to `Tasks`.

C. Mapping a TIM to the Development Process

The mapping between elements in the TIM and development artifacts must be clear – it needs to be obvious which work products created during development map to which artifact types and which elements of the TIM correspond to the different trace link types [24].

An advantage of the *work product-driven, type-focused approach* of TIM II is that it is trivial to map the TIM to the existing artifacts of a development process. Each concept in the TIM corresponds to a particular artifact type (e.g., the `Safety Case` or a traceable unit within an artifact type (e.g., a `Safety Goal` within the `Requirement Specification` package). On the other hand, such TIMs are also more specific to the concrete development process and the work products it creates.

TIM II on the other hand is more abstract and generic and can be adapted to different development processes. This higher level of abstraction also requires an additional document that maps the different work products to their respective roles in the TIM. This characteristic is shared by many TIMs from the literature, especially *role-focused* ones. SafeTIM [8] is similarly independent of concrete work products. The TIM for RAMS in [29] is more specific than SafeTIM since it contains the outcomes of the safety activities (it is thus *work-product focused*), but less specific than TIM I since it does not prescribe the types of the artifacts (and is thus *role-focused*). Arguably, using TIMs from literature for comparison introduces a bias since these TIMs are usually more abstract in order to have higher generalizability.

Such abstract TIMs need to be mapped to the concrete artifact types of the development process using a separate document or a refinement. This mapping document needs to describe which concrete work product maps to which artifact type, which trace link types exist, and how non-traceable artifacts (e.g., the `Storage Location` in TIM I) are resolved.

VII. Conclusion

In this paper, we provide insights into critical design decisions for TIMs and their drivers. Based on an analysis of the design of two different TIMs for safety-critical automotive systems, we have identified that *fitness for purpose* is the main driver of TIM design and that important decisions include the design approach, the artifact focus, the coverage of artifacts, and others. In addition, we have identified a number of important criteria for the comparison of TIMs based on a lightweight systematic literature review and analyzed four TIMs based on that catalogue. We also discuss the impact our findings have on the design of TIMs, their evolution, and their mapping to the development process.

We believe that, even though literature provides a number of specific TIMs, much more insight into TIM design is needed. In particular, in situations in which traceable information spans teams or even organizations (see, e.g., [3], [53]), current guidelines are insufficient. In addition, how and when to evolve TIMs and the traces based on them needs to be investigated further. Finally, we aim to investigate which implications design decisions for the TIM have on working with the trace links in daily practice.

Acknowledgements

This research has received funding from the *Federal Ministry of Education and Research* (BMBF) under grant agreement 01IS18057G and by Vinnova under grant agreement 2018-02228 as part of the ITEA 3 project *PANORAMA* as well as by the Vinnova project *CASUS*.

References

[1] A. von Knethen and B. Paech, "A Survey on Tracing Approaches in Practice and Research," Fraunhofer IESE, Kaiserslautern, Tech. Rep. 095.01/E, 2002.

[2] P. Rempel and P. Mäder, "A Quality Model for the Systematic Assessment of Requirements Traceability," in *IEEE 23rd International Requirements Engineering Conference (RE)*. IEEE, 2015, pp. 176–185.

[3] S. Maro, J.-P. Steghöfer, E. Knauss, J. Horkoff, R. Kasauli, R. Wohlrab, J. L. Korsgaard, F. Wartenberg, N. J. Strøm, and R. Alexandersson, "Managing Traceability Information Models: Not Such a Simple Task After All?" *IEEE Software*, 2021.

[4] M. Rath, D. Lo, and P. Mäder, "Analyzing Requirements and Traceability Information to Improve Bug Localization," in *2018 IEEE/ACM 15th International Conference on Mining Software Repositories*. IEEE, 2018, pp. 442–453.

[5] H. Schwarz, J. Ebert, and A. Winter, "Graph-Based Traceability: A Comprehensive Approach," *Software & Systems Modeling*, vol. 9, no. 4, pp. 473–492, 2010.

[6] P. Mäder, P. L. Jones, Y. Zhang, and J. Cleland-Huang, "Strategic Traceability for Safety-Critical Projects," *IEEE Software*, vol. 30, no. 3, pp. 58–66, 2013.

[7] ISO/TC 22/SC 32, "ISO 26262:2018 – Road Vehicles – Functional Safety," International Organization for Standardization, Standard, 2018. [Online]. Available: https://www.iso.org/standard/68383.html

[8] S. Nair, J. L. de la Vara, A. Melzi, G. Tagliaferri, L. De-La-Beaujardiere, and F. Belmonte, "Safety Evidence Traceability: Problem Analysis and Model," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2014, pp. 309–324.

[9] R. Dardar, B. Gallina, A. Johnsen, K. Lundqvist, and M. Nyberg, "Industrial Experiences of Building a Safety Case in Compliance with ISO 26262," in *2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops*. IEEE, 2012, pp. 349–354.

[10] P. Mäder, O. Gotel, and I. Philippow, "Getting Back to Basics: Promoting the Use of a Traceability Information Model in Practice," in *ICSE Workshop on Traceability in Emerging Forms of Software Engineering*. IEEE, 2009, pp. 21–25.

[11] B. Ramesh and M. Jarke, "Toward Reference Models for Requirements Traceability," *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 58–93, 2001.

[12] O. Gotel, J. Cleland-Huang, J. H. Hayes, A. Zisman, A. Egyed, P. Grünbacher, A. Dekhtyar, G. Antoniol, J. Maletic, and P. Mäder, "Traceability Fundamentals," in *Software and Systems Traceability*. Springer, 2012, pp. 3–22.

[13] P. Letelier, "A Framework for Requirements Traceability in UML-Based Projects," in *Proceedings of the 1st International Workshop on Traceability in Emerging Forms of Software Engineering*, 2002, pp. 30–41.

[14] P. Mäder and J. Cleland-Huang, "A Visual Traceability Modeling Language," in *Model Driven Engineering Languages and Systems*, D. C. Petriu, N. Rouquette, and Ø. Haugen, Eds. Springer, 2010, pp. 226–240.

[15] P. Barbosa, F. Leite, D. Santos, A. Figueiredo, and K. Galdino, "Introducing Traceability Information Models in Connected Health Projects," in *IEEE 31st International Symposium on Computer-Based Medical Systems*. IEEE, 2018, pp. 18–23.

[16] D. S. Kolovos, R. F. Paige, and F. A. C. Polack, "On-Demand Merging of Traceability Links with Models," in *3rd ECMDA Traceability Workshop*, 2006, pp. 47–55.

[17] N. Drivalos, R. F. Paige, K. J. Fernandes, and D. S. Kolovos, "Towards Rigorously Defined Model-to-Model Traceability," in *4th ECMDA Traceability Workshop*, 2008, pp. 17–26.

[18] A. Rodrigues da Silva, "Model-Driven Engineering: A Survey Supported by the Unified Conceptual Model," *Computer Languages, Systems & Structures*, vol. 43, pp. 139–155, 2015.

[19] N. Aizenbud-Reshef, B. T. Nolan, J. Rubin, and Y. Shaham-Gafni, "Model Traceability," *IBM Systems Journal*, vol. 45, no. 3, pp. 515–526, 2006.

[20] B. Vanhooff, S. Van Baelen, W. Joosen, and Y. Berbers, "Traceability as Input for Model Transformations," in *3rd ECMDA Traceability Workshop*. SINTEF, 2007, pp. 37–46.

[21] S. Winkler and J. von Pilgrim, "A Survey of Traceability in Requirements Engineering and Model-Driven Development," *Software & Systems Modeling*, vol. 9, no. 4, pp. 529–565, 2010.

[22] A. Espinoza, P. P. Alarcon, and J. Garbajosa, "Analyzing and Systematizing Current Traceability Schemas," in *2006 30th Annual IEEE/NASA Software Engineering Workshop*. IEEE, 2006, pp. 21–32.

[23] J. Holtmann, J.-P. Steghöfer, M. Rath, and D. Schmelter, "Cutting Through the Jungle: Disambiguating Model-Based Traceability Terminology," in *IEEE 28th International Requirements Engineering Conference (RE)*. IEEE, 2020, pp. 8–19.

[24] M. Taromirad, N. D. Matragkas, and R. F. Paige, "Towards a Multi-Domain Model-Driven Traceability Approach," in *7th International Workshop on Multi-Paradigm Modeling*, 2013, pp. 27–36.

[25] J. Cleland-Huang, B. Berenbach, S. Clark, R. Settimi, and E. Romanova, "Best Practices for Automated Traceability," *Computer*, vol. 40, no. 6, pp. 27–35, 2007.

[26] P. Lago, H. Muccini, and H. Van Vliet, "A Scoped Approach to Traceability Management," *Journal of Systems and Software*, vol. 82, no. 1, pp. 168–182, 2009.

[27] V. Katta, C. Raspotnig, and T. Stålhane, "Presenting a Traceability Based Approach for Safety Argumentation," *Safety, Reliability and Risk Analysis: Beyond the Horizon*, pp. 2037–2046, 2014.

[28] V. Katta and T. Stålhane, "A Conceptual Model of Traceability for Safety Systems," in *Proceedings of the Complex Systems Design & Management Conference*, 2011.

[29] K. Taguchi, S. Daisuke, H. Nishihara, and T. Takai, "Linking Traceability with GSN," in *2014 IEEE International Symposium on Software Reliability Engineering Workshops*. IEEE, 2014, pp. 192–197.

[30] T. Kelly and R. Weaver, "The Goal Structuring Notation – A Safety Argument Notation," in *Proceedings of the Dependable Systems and Networks Workshop on Assurance Cases*, 2004.

[31] A. Agrawal, S. Khoshmanesh, M. Vierhauser, M. Rahimi, J. Cleland-Huang, and R. Lutz, "Leveraging Artifact Trees to Evolve and Reuse Safety Cases," in *2019 IEEE/ACM 41st International Conference on Software Engineering*. IEEE, 2019, pp. 1222–1233.

[32] N. Prat, I. Comyn-Wattiau, and J. Akoka, "Artifact Evaluation in Information Systems Design-Science Research – A Holistic View," *18th Pacific Asia Conference on Information Systems*, vol. 23, pp. 1–16, 2014.

[33] J. Birch, R. Rivett, I. Habli, B. Bradshaw, J. Botham, D. Higham, P. Jesty, H. Monkhouse, and R. Palin, "Safety Cases and Their Role in ISO 26262 Functional Safety Assessment," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2013, pp. 154–165.

[34] E. Denney and G. Pai, "Tool Support for Assurance Case Development," *Automated Software Engineering*, vol. 25, no. 3, pp. 435–499, 2018.

[35] Uber Technologies Inc. Our Safety Case. Uber's Commitment to Self-Driving Safety. https://uberatgresources.com/safetycase. Accessed: 2021-02-22.

[36] S. Kokaly, R. Salay, M. Chechik, M. Lawford, and T. Maibaum, "Safety Case Impact Assessment in Automotive Software Systems: An Improved Model-Based Approach," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2017, pp. 69–85.

[37] K. Attwood, P. Chinneck, M. Clarke, G. Cleland, M. Coates, T. Cockram, G. Despotou, L. Emmet, J. Fenn, B. Gorry *et al.*, "GSN Community Standard Version 1," *Origin Consulting (York) Limited*, 2011.

[38] C. Cârlan, T. A. Beyene, and H. Ruess, "Integrated Formal Methods for Constructing Assurance Cases," in *2016 IEEE International Symposium on Software Reliability Engineering Workshops*. IEEE, 2016, pp. 221–228.

[39] A. Hamann, D. Dakshina, F. Wurst, I. Sañudo, N. Capodieci, P. Burgio, and M. Bertogna, "WATERS 2019 – Industrial Challenge," 10th International Workshop on Analysis Tools and Methodologies for Embedded and Real-Time Systems (WATERS), 2019. [Online]. Available: https://www.ecrts.org/archives/fileadmin/WebsitesArchiv/ecrts2019/waters/waters-industrial-challenge/index.html

[40] J.-P. Steghöfer, B. Koopmann, J. S. Becker, I. Stierand, M. Zeller, M. Bonner, D. Schmelter, and S. Maro, “The MobSTr Dataset: Model-Based Safety Assurance and Traceability,” Jun. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.4981481

[41] Y. Papadopoulos, I. Sorokos, and J. Reich, “ODE Profile V2,” 2019. [Online]. Available: http://www.deis-project.eu/fileadmin/user_upload/DEIS_Open_Dependability_Exchange_Profile_V2_Whitepaper.pdf

[42] S. Maro and J.-P. Steghöfer, “Capra: A Configurable and Extendable Traceability Management Tool,” in *2016 IEEE 24th International Requirements Engineering Conference (RE)*. IEEE, 2016, pp. 407–408.

[43] A. Seibel, S. Neumann, and H. Giese, “Dynamic Hierarchical Mega Models: Comprehensive Traceability and Its Efficient Maintenance,” *Software & Systems Modeling*, vol. 9, no. 4, pp. 493–528, 2010.

[44] J. Cleland-Huang, “Traceability in Agile Projects,” in *Software and Systems Traceability*. Springer, 2012, pp. 265–275.

[45] N. Drivalos, D. S. Kolovos, R. F. Paige, and K. J. Fernandes, “Engineering a DSL for Software Traceability,” in *International Conference on Software Language Engineering*. Springer, 2008, pp. 151–167.

[46] B. Grammel and S. Kastenholz, “A Generic Traceability Framework for Facet-Based Traceability Data Extraction in Model-Driven Software Development,” in *Proceedings of the 6th ECMFA Traceability Workshop*. ACM, 2010, pp. 7–14.

[47] A. Seibel, R. Hebig, and H. Giese, “Traceability in Model-Driven Engineering: Efficient and Scalable Traceability Maintenance,” in *Software and Systems Traceability*. Springer, 2012, pp. 215–240.

[48] A. A. Anda and D. Amyot, “Traceability Management of GRL and SysML Models,” in *Proceedings of the 12th System Analysis and Modelling Conference*, 2020, pp. 117–126.

[49] P. Rempel, P. Mäder, and T. Kuschke, “An Empirical Study on Project-Specific Traceability Strategies,” in *21st IEEE International Requirements Engineering Conference (RE)*. IEEE, 2013, pp. 195–204.

[50] P. Rempel, P. Mäder, T. Kuschke, and J. Cleland-Huang, “Mind the Gap: Assessing the Conformance of Software Traceability to Relevant Guidelines,” in *Proceedings of the 36th International Conference on Software Engineering*. ACM, 2014, pp. 943–954.

[51] M. A. G. Carrión, M. D. Paredes, P. H. Niquen, and Á. F. del Carpio, “Traceability Information Model for Very Small Entities With ISO/IEC 29110,” in *Proceedings of the International Conference on Geoinformatics and Data Analysis*. ACM, 2018, pp. 186–191.

[52] O. Gotel, J. Cleland-Huang, J. H. Hayes, A. Zisman, A. Egyed, P. Grünbacher, A. Dekhtyar, G. Antoniol, and J. Maletic, “The Grand Challenge of Traceability (v1.0),” in *Software and Systems Traceability*. Springer, 2012, pp. 343–409.

[53] S. Seedorf, K. Nordheimer, and S. Krug, “STraS: A Framework for Semantic Traceability in Enterprise-Wide SOA Life-Cycle Management,” in *2009 13th Enterprise Distributed Object Computing Conference Workshops*. IEEE, 2009, pp. 212–219.

[54] M. Taromirad and R. F. Paige, “Agile Requirements Traceability Using Domain-Specific Modelling Languages,” in *Proceedings of the 2012 Extreme Modeling Workshop*, 2012, pp. 45–50.