DEIS Requirements Report

White Paper

www.deis-project.eu

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732242 (DEIS).

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Long Version
CPS	Cyber Physical System
B2B	Business To Business
B2C	Business To Customer
EHR	Electronic Health Record
ODE	Open Dependability Exchange Metamodel
SoA	Service-oriented Architecture
DDI	Digital Dependability Identity
SACM	Structured Assurance Case Metamodel
GSN	Goal Structuring Notation
CAE	Claims, Arguments and Evidence
SoS	System of Systems
MBSA	Model-based Safety Analysis
MBDA	Model-based Dependability Analysis
RAMS	Reliability, Availability, Maintainability, Safety

Authors

Name, Partner	E-mail
Marc Zeller, Siemens AG	Marc.zeller@siemens.com
Ioannis Sorokos, University of Hull	I.Sorokos@hull.ac.uk
Cem Kaypmaz, AVL Turkey	cem.kaypmaz@avl.com
Simone Longo, GM	Simone.longo@gm.com
Eoin O'Carroll, Portable Medical Technology	eoin@portablemedicaltechnology.com
Daniel Schneider, Fraunhofer IESE	daniel.schneider@iese.fraunhofer.de
Jan Reich, Fraunhofer IESE	jan.reich@iese.fraunhofer.de
Joe Guo, Siemens AG	Joe.guo@siemens.com
Christof Kaukewitsch, Siemens AG	Christof.kaukewitsch@siemens.com

1 Executive Summary

The open and cooperative nature of Cyber-Physical-Systems (CPS) poses a significant new challenge in assuring dependability. The DEIS project addresses these important and unsolved challenges by developing technologies that form a science of dependable system integration. In the core of these technologies lies the concept of a Digital Dependability Identity (DDI) of a component or system. A DDI is an assurance case package for a component modelled using the Structured Assurance Case Metamodel (SACM)[1]. DDIs are composable and executable in the field facilitating (a) efficient synthesis of component and system dependability information over the supply chain and (b) effective evaluation of this information in-the-field for safe and secure composition of highly distributed and autonomous CPS.

This document provides an overview and summary on the requirements collected in the DEIS project. Due to the characteristics of the DEIS, the requirements were collected within the three innovation cycles focusing on

- Open framework to efficient exchange dependability-related information over the supply chain of CPS
- Semi-automated synthesis of dependability assurance cases and DDIs
- Fully automated synthesis and evaluation of dependability assurance cases and DDIs applicable to CPS certification in the field

Overall 60 requirements have been collected. These requirements constitute the basis for the development carried out within the DEIS project. Therefore, we show which of the requirements are relevant for which of the DDI engineering stories, and for which of the industrial use cases of the DEIS project. The DEIS concepts developed based on the requirements listed in this document are described as public project deliverables on the DEIS project website[2]. The implementation of the concepts is partly available as open-source code on Github[3].

This document includes:

- An introduction for the DEIS project Work package 2 “Project Requirements”
- Terms and definitions needed to understand the DDI requirements
- Description of the DDI Engineering Stories defined in DEIS Work package 3 “Model Concept”
- Brief Description of the DEIS industrial use cases defined
- The set of project requirements collected in the three innovation cycles of the DEIS project
- Mapping of requirements to the DDI Engineering Stories and to the industrial use cases

[1] http://www.omg.org/spec/SACM/ [Online: 30.05.2019]
[2] http://www.deis-project.eu/dissemination/ [Online: 30.05.2019]
[3] https://github.com/DEIS-Project-EU [Online: 30.05.2019]

2 Introduction

2.1 Overview and Scope

The open and cooperative nature of Cyber-Physical-Systems (CPS) poses a significant new challenge in assuring dependability. The DEIS project addresses these important and unsolved challenges by developing technologies that form a science of dependable system integration. At the core of these technologies lies the concept of a Digital Dependability Identity (DDI) [1] of a component or system. A DDI is an assurance case package for a component modelled using the Structured Assurance Case Metamodel (SACM) [2]. DDIs are composable and executable in the field facilitating (a) efficient synthesis of component and system dependability information over the supply chain and (b) effective evaluation of this information in-the-field for safe and secure composition of highly distributed and autonomous CPS.

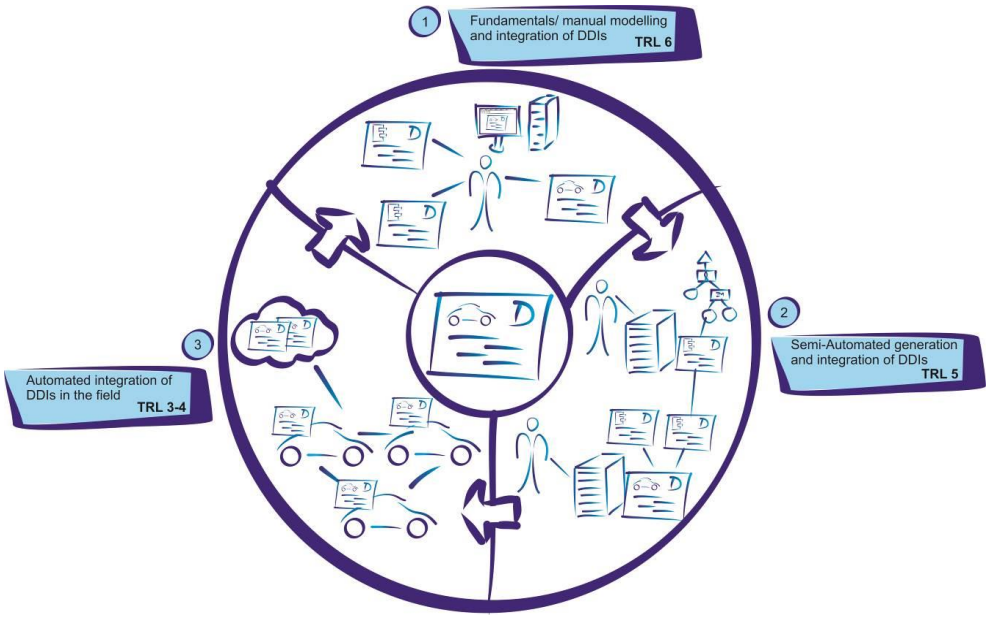

Figure 1: DEIS Innovation Cycles

As illustrated in Figure 1, the DEIS project is implemented in three innovation cycles. The first innovation cycle (1) focuses on the fundamentals including the definition of appropriate formats of DDIs as well as the Open Dependability Exchange Metamodel (ODE). As a first step towards the dependability collaboration workspace, existing tools are extended with interfaces to the workspace based on the ODE. Moreover, the first tools for the modelling of DDIs and ODE-compliant dependability models are created, so that a loose coupling between different companies and their tools along the value chain becomes possible. The second innovation cycle (2) then focuses on the semi-automated generation of DDIs as well as semi-automated analyses based on DDIs. Based on the research results of this innovation cycle,

component providers can generate a DDI for their component based on their existing dependability documentation, independently of which tool or methodology they use. The component integrator can then include the components' DDIs to compile the dependability assurance case for his system. Semi-automated means in this context that we expect that an engineer has to manually provide some additional information in order to enable this transformation, since off-the-shelf tools contain all required dependability-related information. The third innovation cycle (3) then considers the automated integration of decentralized systems of systems during field operation. In principle, this step follows a similar approach like the previous innovation cycle, but a higher degree of automation is required.

2.2 Goals of the Deliverable

This deliverable provides an overview and summary on the requirements collected in the DEIS project as a result of Work Package 2. This deliverable includes the specified requirements for an open framework to efficiently exchange dependability-related information over the supply chain of CPS, the requirements for the semi-automated synthesis of dependability assurance cases and DDIs, and the requirements for the fully automated synthesis and evaluation of dependability assurance cases as well as DDIs applicable to CPS certification in the field. These requirements defined in this deliverable have been used as input for the technology development as well as the application development in the DEIS project. Moreover, the results of the evaluation of the DEIS project outcome are checked against the project requirements.

2.3 Document Structure

The document is organized as follows: In Section 3 we will explain a set of terms and definitions which are important for the DEIS project. In Section 4 an overview of the DDI engineering stories representing concrete challenges that engineers face during the engineering lifecycle activities of a dependability-critical system or system-of-systems. Afterwards, of the industrial use cases considered in the DEIS project are outlined. Based on the DDI engineering stories and the industrial use cases a set of requirements for DDIs are derived in Section 6. Section 7 presents the mapping of requirements to the DDI Engineering Stories and to the industrial use cases.

3 Terms & Definitions

List of terms and definitions used in the project

Term	Description
Dependability	According to [3] Dependability includes the following attributes: • Availability • Reliability • Safety • Confidentiality • Integrity • Maintainability
Availability	Readiness for correct service
Reliability	Continuity of correct service
Safety	Absence of harmful consequences on the user(s) and the environment
Confidentiality	The absence of unauthorized disclosure of information
Integrity	Absence of improper system alteration
Maintainability	Ability for a process to undergo modifications and repairs
Component	A component represents a modular part of a system, that encapsulates its content and whose manifestation is replaceable within its environment. A component defines its behaviour in terms of provided and required interfaces.
System	A system is an entity that interacts with other entities, i.e., other systems, including hardware, software, humans, and the physical world with its natural phenomena. These other systems are the environment of the given system. The system boundary is the common frontier between the system and its environment.
System of systems	System of systems is a collection of task-oriented or dedicated systems that pool their resources and capabilities together to create a new, more complex system which offers more functionality and performance than simply the sum of the

Term	Description
	constituent systems.
Digital Dependability Identity (DDI)	An assurance case package for a component/system modelled using SACM as well as the ODE product model.
Open Dependability Exchange (ODE)	Metamodel encompassing both SACM and the product meta-models, allowing the creation of machine processable DDIs.
Executable DDI	A DDI that has strong semantics which enable interpretation and analysis to be done during operation. The outcome of this analysis could determine whether a system can safely participate in a system of systems or whether a set of systems can safely cooperate at a particular moment in time.

Table 1: Definition of terms

4 DDI Engineering Stories

In this section, we link the set of use case scenarios to the engineering stories in which DDIs are used.

The engineering story concept is introduced in more details in Deliverable D3.1 [4] and aims to represent concrete challenges that engineers face during the engineering lifecycle activities of a dependability-critical system or system of systems. The following subsections give an overview of the currently existing engineering stories. Over the course of the project, the partners will prioritise the most important stories as candidates to be supported directly by the ODE. All other engineering stories shall serve as a reference to demonstrate that DDIs can be leveraged to support future engineering challenges not fully addressed in the DEIS project. Note that the list of engineering stories is still not complete at this point of the project as not all the industrial use cases have been analysed in full details for the DDI improvements involved yet.

4.1 ES1: Multi-tool interoperability for dependability artefact exchange

This engineering story is focusing on the variety of tools to model and analyse the dependability aspect of systems or systems of systems. Although the modelling and analysis techniques used are often similar to each other, it is currently not possible to easily exchange dependability models between different tools. In order to solve this challenge, the DEIS approach shall capture the essence of dependability-related techniques to achieve a common interface for dependability model and information exchange between different tools at design time. This idea is illustrated in Figure 2, where several techniques (CBD=Component-Based Design, CFT=Component Fault Tree, GSN=Goal Structuring Notation, FMEA=Failure Mode and Effect Analysis) are supported by different tools.

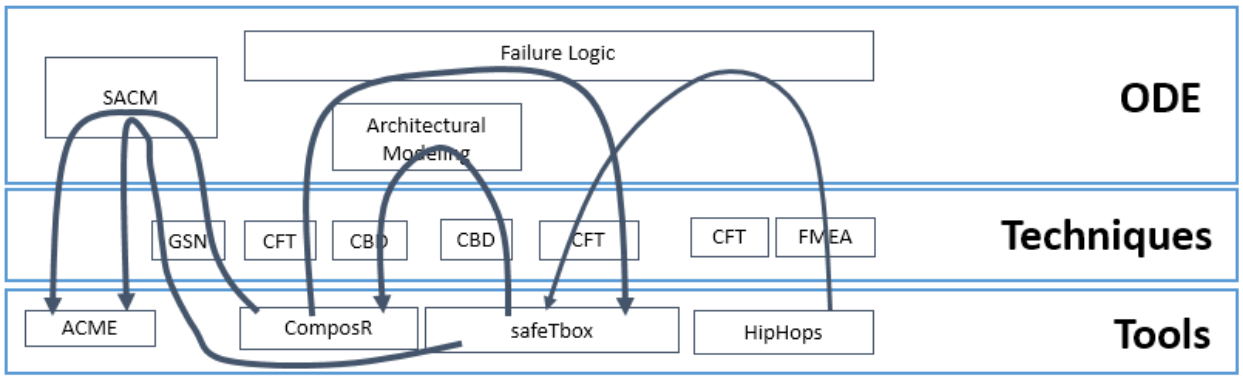

Figure 2: Multi-tool interoperability with the help of the ODE as exchange format

4.2 ES2: Protection of intellectual property in distributed development scenarios

ES2 is focusing on the distributed fashion in which closed and open embedded systems are developed.

As depicted in Figure 3, different kinds of distribution are possible: (a) distribution in a single company, (b) distribution among integrator and supplier companies or (c) distribution "@runtime", where running

systems exchange models and the manufacturer companies of these systems might not even be in contact with each other. Depending on the distribution scenario, different levels of detail have to be exposed to the recipients of the model to enable them to perform the intended task with the models they received. In these cases, the DDI concept can help by providing ODE meta-model packages suitable for the different scenarios on the one hand, and (semi-)automated algorithms that help in producing and consuming the exchanged information in a goal-oriented manner on the other hand.

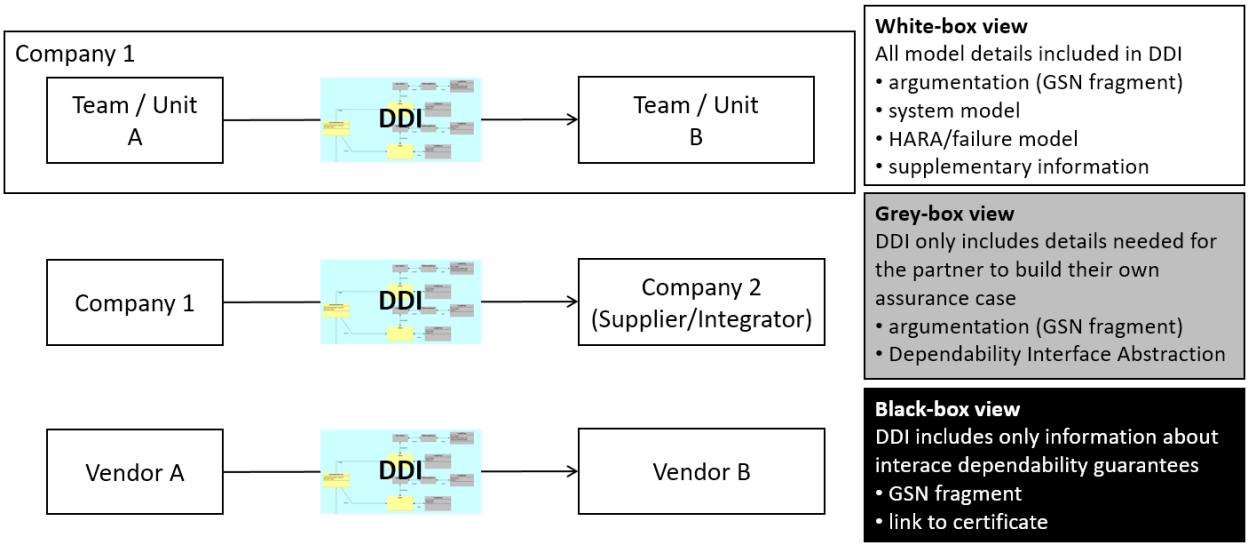

Figure 3: Different levels of details exchanged in different scenarios

4.3 ES3: Integration of safety case fragments into a system safety case

When several subsystems are to be integrated into a system together, a system safety case providing a sound argumentation of why the overall system is safe has to be created. Therefore, the modular safety arguments (also called fragments here) of the subsystems have to be combined and additional argumentation has to be added, taking into account the integration context of the overall system, which was only assumed during the development of the subsystems. The challenges to be solved within ES3 are: (a) providing means to model and exchange *modular* safety case fragments, (b) formalising relevant parts of the safety argumentation content and structure, and (c) providing (semi-)automated algorithms capable of helping the integration engineer to check the consistency of the interface.

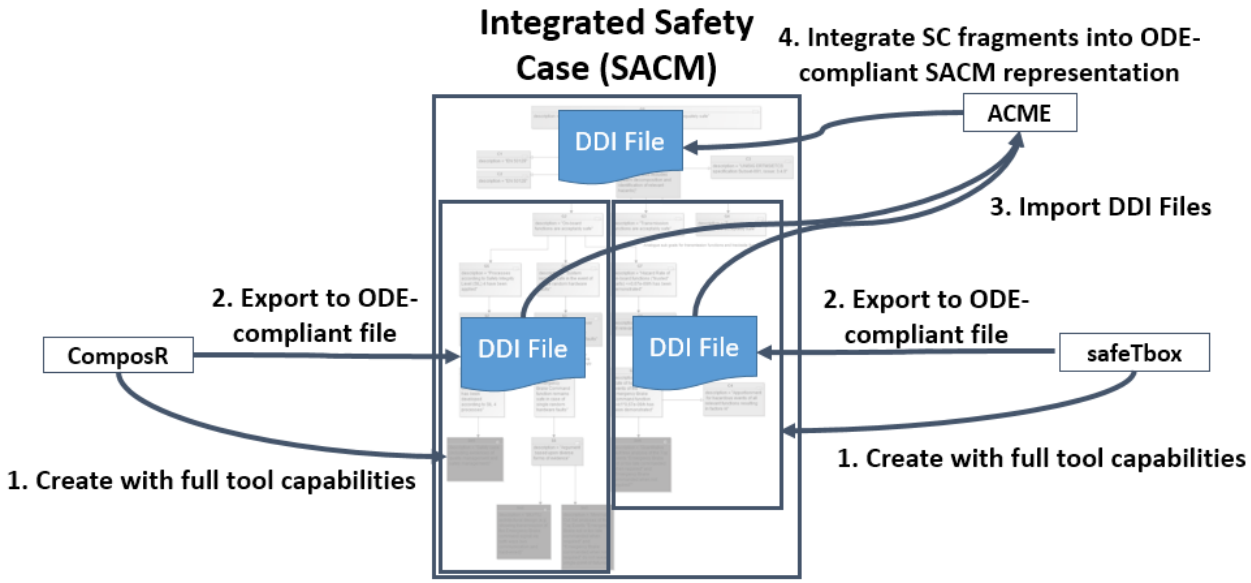

Figure 4: Safety case integration with the help of DDIs

4.4 ES4: Failure interface compatibility matching during design time system integration

A similar problem as in "ES3: Integration of safety case fragments into a system safety case", exists regarding the synthesis of system and failure logic models from a set of subsystems. A core issue during the integration is checking for compatibility between the system and the failure interface of integrating and integrated systems. In this case, compatibility is not only characterised by the explicit matching of exchanged information (e.g., data flows or failure propagation), but also by matching assumptions with expectations related to the interface. DDIs shall here assist in these tasks by providing capabilities for describing formalised system and failure interfaces as well as common-cause-related aspects of the sub-systems.

4.5 ES5: Trade-off support for RAMS property optimisation at design time

This engineering story is embedded into the scenario that an integrator company has to integrate components provided by different suppliers. The specific challenge targeted in this ES is that nowadays, there is little support for performing trade-offs and optimisation among the RAMS (=Reliability, Availability, Maintainability, Safety) properties when considering all properties at once. The idea for solving this issue is to support the integration engineer in a (semi-)automated way by using DDIs so that qualitative and quantitative trade-offs with respect to the RAMS properties are facilitated and the process is accelerated.

4.6 ES6: Synthesis of dependability runtime models for safe system of systems integration at runtime

ES6 focuses on the newest generation of embedded systems; which are developed with the goal that they shall be fit for dynamic participation in different collaboration scenarios with other systems. In such cases, the manufacturers of individual systems do not have complete knowledge about all the details of the future collaboration scenarios and collaboration partner systems at design time. In order to develop a system that can successfully participate in many different collaboration scenarios at runtime, a set of variants have to be built into the product. In this sense, the knowledge of whether the quality of a received signal or service from another system is good or bad can only be resolved at runtime, and therefore the final integration of collaborative systems can only be carried out at runtime as well.

As the runtime integration has to be carried out in a fully automated manner, the models have to be formalised to a sufficient degree. Especially regarding the assurance of dependability for systems of systems, models have to be created to capture certain behavioural and structural variants (also known as configurations) together with the assumptions, demands and guarantees describing dependability-related properties of the system.

4.7 ES7: Modelling of security aspects and safety analysis with respect to malicious reasons

When open systems of systems are to be built that communicate with each other over open communication channels, the assurance of security is of major importance to prevent dependability problems caused by malicious reasons (e.g., hacker attacks). Particularly the dependability attributes confidentiality, integrity and availability (CIA) can be affected by malicious faults. However, malicious faults can also affect reliability or safety. In current development processes for embedded systems, the modelling and analysis of security does not have a long tradition yet and is therefore still loosely integrated with other disciplines. In order to reduce the overhead times and make the interfaces between security models, safety models and system design models explicit, DDIs can help by incorporating security-related models that are explicitly related to the models created by other disciplines in order to enable integrated reasoning on security.

5 Industrial use Cases

In this Section, we present the industrial use cases which are used in the DEIS project to apply DDI concepts developed in the DEIS project in the application domains automotive, railway and healthcare,

5.1 Industrial use case 1 (automotive): Intelligent physiological parameter monitoring for automotive applications to monitor driver comfort and capability to safely control the vehicle

The aim of this use case is the introduction of a dependable physiological monitoring system applied to a connected vehicle, capable to identify physiological parameters (heart rate and respiration rate) and to evaluate the health of driver and other occupants.

The proposed environment contains a comprehensive package of technologies, tools and services that support the drive session in evaluating the health status and can take safety action to minimize the risk of injuries.

The camera system adopted is based on a Single-Photon Avalanche Diode (SPAD) technology [5] capable to extract physiological parameter form the imaging acquisition, analysing volumetric measurements (plethysmography, PG) of the skin, and changes in skin volume can be optically detected through photo plethysmography (PPG).

All the acquired data are processed with complex techniques of machine learning capable to extract the passenger's health condition, detecting acute illness and drowsiness or any other emergency condition that can be a potential risk factor for the occupants. The evaluation of the health risk level enables particular features on the vehicle, that takes control over functions. For instance, through autonomous driving, the vehicle can move to the emergency lane or the nearest safety area, or in case of high danger, drive the vehicle to the Hospital / First Aid Services.

The same acquired parameters are also continuously transmitted to a cloud-based system and constantly monitored. The target is to provide an on-demand medical check-up with online support, leveraging on GM's current cloud infrastructure. In case of an emergency condition, the online support shall evaluate the risk level associated with the condition, and remotely take the action to inform promptly the Emergency/First Aid Services, relaying the overall health condition of each passenger on-board, allowing healthcare support to be prepared in advance.

In case of an accident, in addition to the health information of the occupants always provided before the accident, the system communicates the vital signs of people that are still detectable after the impact, until rescue arrival.

At the same time, this system introduces the threat of a new security attack. One of the challenges of this use case will be to assure the *Confidentiality* of health data transmitted, and guarantee *Safety* and *Plausibility* analysis on acquired signals, as well as *Integrity* on communication among different CPSs. The

impact of cyber security on this kind of system is one of the most challenging aspects that the DDI approach can help address.

5.2 Industrial use case 2 (automotive): Evaluation of platooning functions and the dependability impact of truck platooning on highway

AVL changed the use case during the second year. Truck platooning is a new use case which will increase the impact of DDI concept in automotive industry. Platooning is collaboration between vehicles which allow vehicles to follow each other closely by sharing information. The aim is to automate lateral and longitudinal control of the vehicles which follow the lead vehicle, in order to save significant amounts of energy, by reducing aerodynamic drag force due to decreasing the distance between vehicles. This will have the effect of reducing fuel consumption, CO2 emissions and increasing highway capacity. Also, as the following vehicles can brake immediately, with zero reaction time, it has the potential to improve safety. Platooning technology reduces the stress of stop-and-go driving. It is an advanced form of Cooperative Adaptive Cruise Control.

The truck platooning is creating a platoon to reduce time gaps between the trucks in order to increase energy efficiency, improve safety and reduce truck driver loads. But the communication between the trucks for increased efficiency and safety comes with increased security threats. In platooning scenarios, guaranteeing the systems' dependability requirements poses new challenges. Platooning is also produced by various stakeholders in the value chain (such as OEMs, suppliers, etc.) and, therefore, safety information about components and subsystems need to be interoperable and exchangeable.

Platooning function for heavy duty trucks is one of the main AD function highly depends on V2V communication. In SAE L4 platoon function, platoon level decisions are taken by the platoon leader and broadcasted to the follower vehicles and executed by each member without any need of a human driver or operator in a constrained operation boundary. Two-way information flow between vehicles and different communication topologies between members creates wide range of dependability. Each platoon member has its own on-board sensors and V2V information for environmental measurements, therefore this overlapping information must be examined considering DDI's and decision whether or not to accept the V2V information must be taken respectively. Additionally, execution of tasks that are assigned by leader vehicle is required to be checked in terms of security and safety by the followers. To achieve that, Model Predictive Control framework based novel control strategies for inter-vehicular distance control will be developed and integrated to simulation environment. For this highly connected use case, overall and individual platoon efficiencies will be examined for energy predictive functions utilizing traffic and road topology. Besides, platoon stability will be examined in terms of vehicle dynamics and communication characteristics limitations. Lastly dependability assessment in terms of safety will be implemented.

The use-case is to develop cooperative control algorithms for the platooning of trucks using AVL's vehicle dynamics tools and Simulink. In order to further enhance and develop platooning control algorithms, the truck is modeled on AVL VSM^4 for the comprehensive evaluation of complex platooning algorithms. The use-case environment contains an extensive package of tools and services that support 'in the vehicle

modeling' and enables improvement of various vehicle attributes from the initial concept to the testing phase. The use-case environment allows for further simulation such as long platoons, various scenarios like intervening vehicle, vehicles with different load and load type capabilities, handover the control from platooning to ACC and ACC to platooning in the case of a communication loss or in an emergency situation.

5.3 Industrial use case 3 (railways): Dependability framework with heterogeneous (cross-industry) participants

The European Train Control System (ETCS) provides standardized train control in Europe and eases travelling with trains crossing the borders of all countries in Europe. Thus, enabling "plug & play" for railway systems scenarios as a long-term objective [6]. However, in such "plug & play" scenarios, guaranteeing the systems' dependability requirements poses new challenges. Consequently, certification activities (w.r.t. CENELEC standards EN 50126 [7] & 50129 [8]) require accurate planning and must react quickly to design changes within the system development process. Systems of systems in the railway domain are also produced by various stakeholders in the value chain (such as national or even regional public transport authorities, national safety authorities, railway undertaking, OEMs, suppliers, etc.) and, therefore, safety information about components and subsystems (rolling stock, track-side and railway systems) need to be interoperable and exchangeable.

In this use case, we utilise DDIs to interchange safety-relevant information (including e.g. safety requirements, models, and assessments) during the development life-cycle of the trackside and the on-board ETCS units (ETCS Level 2). Thus, evaluating how the DEIS approach eases the development process in achieving safe and interoperable railway systems.

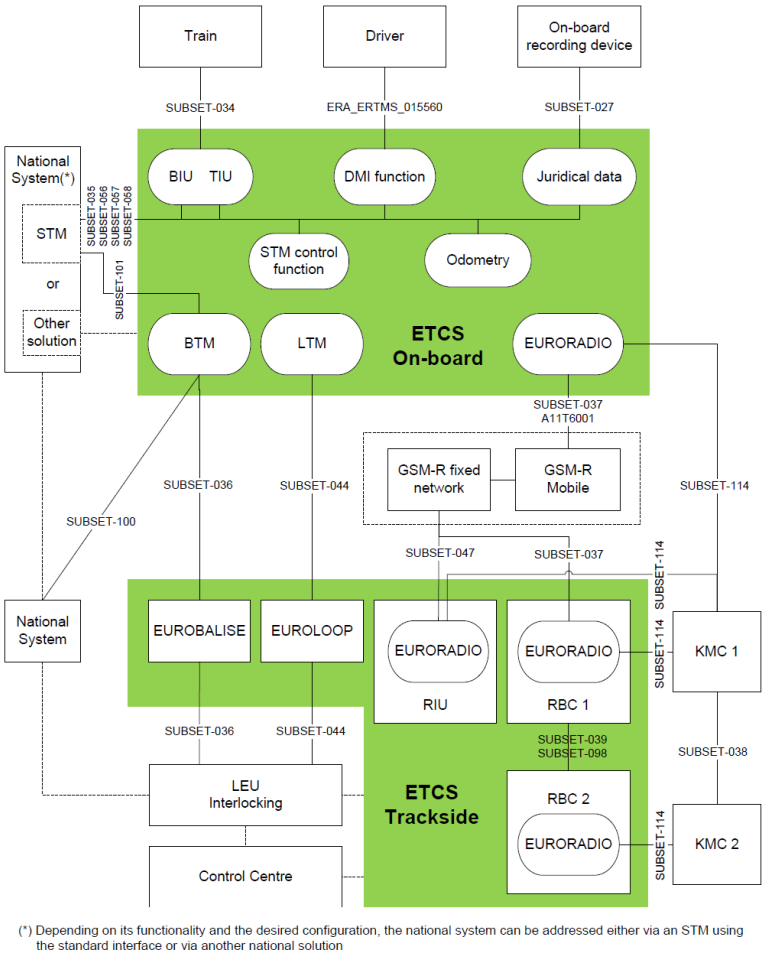

Figure 5: ERTMS/ETCS Reference Architecture

5.4 Industrial use case 4 (healthcare): Clinical decision support app for oncology professionals

The aim of this use case is to describe ad hoc data synchronization among mobile devices, but in particular the ONCOassist app and hospital Electronic Health Records (EHRs), without depending on a central server. This can then be applied to synchronize personal data between two devices on the fly. The ad hoc synchronization needs a discovery module, user authentication module, and application data synchronization module. The result of this is that new clinical information is recorded in an existing patient record in EHR in a way that preserves the intended meaning of the information being transferred.

The primary actors are the mobile app, which connects with an EHR and transmits data to it (a push model). The EHR, which connects with app and asks it to send a data update (pull model). Other actors are health practitioners who use the app and enter or update clinical information about their patients. Practitioners who use the EHR and view (or request, in the pull model) up-to-date clinical information about their patients. The only constraint is that the update and display of new clinical data must occur within 1 second.

The detailed push use case would flow as follows. The mobile app obtains new data. The app is triggered e.g., new patient information is available. App requests to open a connection with EHR. EHR accepts the connection request and a connection is opened. App packages the new clinical information in a format that EHR can understand. App sends the updated clinical data to EHR. EHR validates the updated clinical data. EHR accepts the data transfer. EHR translates the data into its own format and updates the relevant patient record. EHR closes the connection. Practitioner using EHR views the patient record containing the updated clinical data.

An alternative pull use case is as follows. App obtains new data. EHR is programmed to request an update from EHR, or practitioner using EHR manually requests an update. EHR requests to open a connection with EHR, with the request containing a query. App accepts the connection request and a connection is opened. The app packages the updated clinical information in a format that EHR can understand. The app sends the updated clinical data to EHR. EHR accepts the data transfer. EHR translates the data into its own format and updates the relevant patient record. EHR closes the connection.

Potentially, the following failure modes could occur: Data transfer trigger on app does not occur. EHR rejects app request for a connection. App sends the data but EHR can't translate it into its own format. New clinical data from App fails validation by EHR. EHR updates its patient record with the new data but the data definitions don't match, and the data are successfully updated but the meaning has been altered. The connection hangs and doesn't successfully close.

The expected end result is that the EHR and app will have the new clinical data in sync with each other.

For this integration to occur, we use a number of well accepted existing standards – both within healthcare and more generally in the web service development.

These standards are:

- FHIR: Describes the content of data being exchanged, and the query API to be used. FHIR is Health specific.
- OAuth2: An Authorization framework widely used outside of health – particularly allowing users to permit client applications (like mobile and web) to access their data. It permits the authentication (are you who say you are) and the authorization (what are you allowed to to) processes to be delegated to separate 'Authorization Servers' so that the application supplying the data (the 'Resource Server') doesn't have to do this – it can delegate that to the Authorization server.
- OpenID: Built on top of OAuth2, this supports user identity – personal information about the user to be given to the client application.

6 DEIS Project Requirements

In this section, we present the requirements collected during the three innovation cycles of the DEIS project. The requirements are the basis for the DEIS project.

Due to the characteristics of DEIS, the requirements were collected within the three innovation cycles focusing on

- Open framework to efficient exchange dependability-related information over the supply chain of CPS in the 1st innovation cycle
- Semi-automated synthesis of dependability assurance cases and DDIs in the 2nd innovation cycle
- Fully automated synthesis and evaluation of dependability assurance cases and DDIs applicable to CPS certification in the field in the 3rd innovation cycle

In the following, the requirements defined in the 1st innovation cycle are marked with green and focus on the open framework to exchange dependability information, the tooling support and infrastructure to collect dependability information, and the creation, modification, and the exchange DDIs. The requirements defined in the 2nd innovation cycle are marked with yellow and refine and extend the project requirements defined in the 1st innovation cycle. The requirements defined in the 3rd innovation cycle focus on the DDI at runtime and are marked with blue.

The requirements are further grouped according to their topic as follows:

- General requirements (Section 6.1),
- Requirements regarding the DDI content (Section 6.2),
- Requirements regarding the exchange of DDIs (Section 6.3),
- Requirements regarding the DDI tooling (Section 6.4), and
- Requirements regarding the runtime DDI (Section 6.5).

In total, 60 requirements have been collected and serve as the basis to the development in DEIS project. Thereby, 31 have been define din the 1st innovation cycle, 19 in the 2nd innovation cycle and 10 in the 3rd innovation cycle.

6.1 General requirements

Requirement	ID: RQ_001	
Name	Machine-readable representation	
Category	Functional	
Description	DDIs need to be machine readable in order to exchange and evaluate DDIs between different systems. Thus, an exchange format for runtime DDIs for runtime evaluation is required	

Dependencies	-
Conflicts	-

Requirement	ID: RQ_002	
Name	Composability	
Category	Functional	
Description	DDIs must be composable, so that DDI of a system / sub-system / system-of system can be derived from the DDIs of its individual elements.	
Dependencies	RQ_002_01, RQ_002_02	
Conflicts	-	

Requirement	ID: RQ_002_01	
Name	Modularity	
Category	Functional	
Description	DDIs shall be created and exchanged in a modular way, where the DDIs can be combined into compound DDIs.	
Dependencies	RQ_002	
Conflicts	-	

Requirement	ID: RQ_002_02	
Name	Openness	
Category	Functional	
Description	DDIs shall be transparent and accessible to other related DDIs, in order to enable data exchange between DDIs. Openness in this context means that well-defined interfaces for data exchange and information hiding are available. This ensures (1) that all the information in the DDI is accessible by a consumer (e.g. human/machine) and (2) it is possible to define mechanisms to also hinder the access to certain consumers.	
Dependencies	RQ_002	
Conflicts	-	

Requirement	ID: RQ_003	
Name	Independence from tools	
Category	Functional	
Description	DDIs need to be independent from any tool or tool-specific format used during the development and operation to create information. Hence, information must be stored in a format which is independent from any tool-specific format. DDI data elements shall be sufficiently abstract to represent data structure, their belonging data attributes and relationships / information flow between such elements.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_004	
Name	Compliance with safety standards	
Category	Non-Functional	
Description	DDI shall represent an assurance case which is compliant with the specific safety / RAMS standards of the addressed application domains (e.g. IEC 61508 for safety requirements, ISO 26262 in the automotive domain, and EN5012x railway domain etc.). Hence, it should be possible to generate the respective documentation needed for a compliant assurance case at least in a semi-automated way. DDI should facilitate generating the required artefacts according to relevant standards. The definition and elicitation of DDI data elements should conform to the lifecycle of the relevant standards. DDI shall be verifiable and testable against its functional and dependability requirements. In such cases DDIs shall include quantified dependability properties such as failure probability for plausibility checks during runtime.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_005	
Name	Parameterization	
Category	Functional	
Description	DDI suppliers or consumers must be able to set specific parameters in order to instantiate a DDI in a specific context (e.g. temperature, maintenance parameter, etc.) when a DDI is integrated into a system or system-of-systems. By using parameterization, a supplier can specify a DDI for a component or system which may be used by others in various contexts. Therefore, a DDI must store information about possible parameter or variation functions of specific information. Moreover, it shall be possible that parameters are set (semi-)automatically e.g. depending on certain environmental or operational conditions so that the user is relieved from standard activities and can concentrate on more sophisticated actions or decisions.	
Dependencies	RQ_009_03	
Conflicts	-	

Requirement	ID: RQ_006	
Name	System Responsiveness	
Category	Non-functional	
Description	Depending on the context, the system shall react properly activating predefined DDI actions within expected time constraints. Only activities with relevance for the user shall be prompted. This may for instance comprise activities requiring user acknowledgement or interaction. Other activities shall be performed (semi-) automatically but shall be documented and traceable.	
Dependencies	-	
Conflicts	RQ_012, RQ_013, RQ_014	

Requirement	ID: RQ_007	
Name	Robustness	
Category	Non-functional	

Description	System shall guarantee the DDI content in the defined operative and environmental conditions
Dependencies	-
Conflicts	-

Requirement	ID: RQ_008	
Name	Representation of the current system condition	
Category	Non-functional	
Description	DDI shall be representative of the condition of the system during runtime (should be able to document environmental/contextual information)	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_022	
Name	Traceability and version control	
Category	Non-Functional	
Description	The changes of the DDI shall be recorded for traceability and version control. (ODE should provide means to document versions)	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_023	
Name	Glossary and Terminology	
Category	Functional	
Description	To avoid ambiguities, DDIs shall have a glossary of the data elements according to relevant standards and usage scenarios. Naming rules shall be utilized.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_024	
Name	Handbook and comments	
Category	Functional	
Description	A DDI handbook shall be created for functionality of DDI and explanation of the data elements, their relationships, elicitation stage, usage limitation, usage assumption, their constraints, known bugs, monitoring mechanisms etc. Comments shall be made to ensure the understandability of the DDI.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_034	
Name	Privacy by Design (GDPR)	
Category	Non-Functional	
Description	DDIs should incorporate organizational and technical mechanisms to protect personal data in the design of new systems and processes; that is, privacy and protection aspects should be ensured by default	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_035	
Name	Data breach notification (GDPR)	
Category	Functional	
Description	DDIs shall keep track of any personal data breach, to allow the regulator and data subject to be informed within 72 hours from the breach event.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_036	
Name	Data portability identification (GDPR)	
Category	Functional	

Description	In case of personal data, a DDI has the accountability to identify and ensure the protection and privacy of personal data when that data is being transferred outside to a third party and / or other entity.
Dependencies	-
Conflicts	-

Requirement	ID: RQ_037	
Name	Data perimeter (GDPR)	
Category	Functional	
Description	GDPR Privacy policy is applied considering the origin of the personal data (from where data comes, and where data has been collected). DDI that manages personal data shall keep track about data origin, to facilitate the regulator to apply data protection policy properly, country by country.	
Dependencies	-	
Conflicts	-	

6.2 Requirements regarding the DDI content

Requirement	ID: RQ_009	
Name	Assurance Case Package	
Category	Functional	
Description	DDIs shall be linked to SACM assurance case packages, including clearly defined assurance assertions.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_025	
Name	Content of DDI Assurance Package	
Category	Functional	
Description	DDI assurance packages shall include both functional (functional requirements, architecture description etc.) and dependability artefacts (dependability requirements,	

	dependability analysis results etc.). The relationships between the functional data elements and their dependability artefacts shall be presented in the DDI assurance metamodel. Such dependencies shall be represented through the whole lifecycle of the DDI by the use of e.g. import/export functions or links/references.
Dependencies	-
Conflicts	-

Requirement	ID: RQ_009_01	
Name	Failure modes	
Category	Functional	
Description	DDIs must represent information about both random and systematic failures. This includes information about failure modes, failure probability (and distribution), MTBF, useful lifetime, parameter-based formulas for failure rate calculation. (Weibull, ...) etc. All the failures modes, which are identified via various dependability analyses, shall be referenced in a failure mode container, through this way the reusability of the identified failure modes / hazards information will be ensured.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_02	
Name	Failure propagation information	
Category	Functional	
Description	DDIs must represent information about the propagation of failures (bottom-up/inductive & top-down/deductive). This can be in form of FME(D)A, CFT, Hip-Hops, Markov Chains etc.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_04	
Name	Failure detection	
Category	Functional	
Description	DDIs must represent information about the failure detection mechanisms provided.	
Dependencies	RQ_009_09	
Conflicts	-	

Requirement	ID: RQ_009_05	
Name	Maintenance data	
Category	Functional	
Description	DDIs must represent information relevant for maintenance operations in the field (e.g. required inspection intervals, maintenance procedures). Hence, DDIs shall be used to support the generation of maintenance artefacts such as maintenance documentation and plans as well as tool and training demand.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_06	
Name	Information to enable diagnosis in the field	
Category	Functional	
Description	DDIs must represent relevant information to enable or to (semi-)automatically perform diagnosis in the field. Diagnosis information may be conditions or models (e.g. in form of Bayesian networks) or warnings It shall be possible to (semi-) automatically derive or event conduct relevant activities, e.g. in order to avoid safety issues or to protect equipment.	
Dependencies	-	

Conflicts	-

Requirement	ID: RQ_009_07	
Name	Dependability-relevant timing information	
Category	Functional	
Description	DDIs must represent information about timing behaviour which is relevant in terms of dependability. For instance, times for start-up, failure diagnosis, Mean Down Time, inspections intervals, reaction times etc.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_08	
Name	Spare parts information	
Category	Functional	
Description	DDIs need to include information about spare parts (e.g. availability), repair times (e.g., if the respective component / sub-component / system is repairable) as well as proven concepts (e.g. storage on site, in a central stock) in order to establish corresponding artefacts. This is additional information beyond the Mean Down Time.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_09	
Name	Fault tolerance	
Category	Functional	
Description	DDIs must represent information about fault tolerance mechanisms. Further, they shall describe failure and exception handling mechanisms.	
Dependencies	RQ_009_04	

Conflicts	-

Requirement	ID: RQ_009_10	
Name	Information about the system architecture	
Category	Functional	
Description	DDIs must represent information about the functional and physical architecture of a component / sub-system / system as far as needed for dependability analysis (e.g. functions, component, interconnections as well as related dependability-relevant information such as integrity level and control and protection paths).	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_009_11	
Name	Results of dependability analyses	
Category	Functional	
Description	DDIs must represent information about the results of performed dependability analyses (e.g. the cut sets of a Fault Tree Analysis as well as underlying assumptions).	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_010	
Name	Confidence in assurance assertions	
Category	Functional	
Description	DDIs must represent information about the confidence in the assurance assertions.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_011	
Name	Assurance case argument shall be given by means of SACM models	
Category	Functional	
Description	Support must be provided for engineers to generate DDIs in the required SACM model format. For example, translating information from MBDA models or ConSerts conditional assertions into SACM elements.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_026	
Name	Qualitative and quantitative data elements	
Category	Functional	
Description	DDIs shall include both qualitative and (if required) quantitative data elements (incl. dependability properties, such as SIL and failure rate etc.).	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_027	
Name	Mapping between tool specific data and DDI	
Category	Functional	
Description	Mapping between the tool-specific data and the ODE metamodel shall be done in order to ensure the modularity and openness of the ODE.	
Dependencies	RQ_002_01, RQ_002_02	
Conflicts	-	

Requirement	ID: RQ_028	
Name	Mapping between DDIs and SACM	
Category	Functional	
Description	Mappings between the DDI data elements (incl. their	

	relationships) and the data elements of SACM shall be performed, in order to create an integrated system assurance case argumentation.
Dependencies	-
Conflicts	-

6.3 Requirements regarding the exchange of DDIs

Requirement	ID: RQ_012	
Name	Authenticity	
Category	Functional	
Description	DDIs containing sensitive information sent through the network shall guarantee the authenticity on all actors involved, senders and receivers, also allowing (semi-) automated processing.	
Dependencies	-	
Conflicts	RQ_006	

Requirement	ID: RQ_013	
Name	Integrity	
Category	Functional	
Description	The DDI shall only be modifiable under defined restrictions and shall use crypto methodologies to avoid tampering and prevent corruption during the transmission of DDIs over the network.	
Dependencies	-	
Conflicts	RQ_006	

Requirement	ID: RQ_014	
Name	Confidentiality	
Category	Functional	
Description	The DDI shall guarantee a certain level of privacy on the data collected and transmitted. Especially, if the DDIs are not evaluated on an embedded device but by a dedicated device (e.g. in the cloud), the	

	transmission of DDI information and the evaluation results must be secured. Moreover, in any situation in which DDIs are interchanged (during development and runtime) the confidentiality of the information (w.r.t. third parties or w.r.t. the provided information/IPs) must be preserved, allowing (semi-) automated processing.
Dependencies	RQ_016
Conflicts	RQ_006

Requirement	ID: RQ_029	
Name	Plausibility checks	
Category	Functional	
Description	Plausibility checks shall be specified by the concept (e.g. to check the content of a DDI). This may, for instance, entail the chosen component or system or claimed dependability properties of components or systems. Plausibility checks shall relate to comparable components or systems or to relevant field experience in corresponding domains.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_030	
Name	Data exchange	
Category	Functional	
Description	DDIs shall enable data exchange between functional and dependability artefacts, such as data exchange between the functional requirements and the functional hazard analysis, the data exchange within each of both fields, such as, for instance, the data exchange between functional requirements and system architecture.	
Dependencies	RQ_001, RQ_002_01, RQ_ 002_02, RQ_003	
Conflicts	-	

Requirement	ID: RQ_031	
Name	Transformation between ODE and tool-dependent data and	

	vice versa
Category	Functional
Description	DDIs shall enable the data exchange between the ODE metamodel repository and the corresponding tools; such as, for instance, the data exchange between the DDI metamodel and the tool for fault tree analysis. The intended tools shall be integrated as modularly as possible in order to ensure tool-independency of DDIs.
Dependencies	-
Conflicts	-

Requirement	ID: RQ_032	
Name	Multi-views	
Category	Functional	
Description	DDIs shall enable creation of multi-views, such as development-time view, runtime view, installation view, maintenance view etc. for different lifecycle phases as well as structural and behavioural views according to the data inherent properties, and further views for various groups such as developer, end user etc.	
Dependencies	-	
Conflicts	-	

6.4 Requirements regarding the DDI tooling

Requirement	ID: RQ_009_03	
Name	Constraints	
Category	Functional	
Description	DDIs must represent and automatically take into account constraints of a component / sub-system / system, such as SRAC (safety-related application constraints) or assumptions and guarantees. Warnings shall be issued automatically in case certain constraints are questionable or may be violated, e.g. during system composition	

Dependencies	RQ_005
Conflicts	-

Requirement	ID: RQ_015	
Name	Synchronization and concurrency	
Category	Functional	
Description	A system / system-of-system (SoS) shall collect all the DDIs of each sub-system / component and keep data in synch for each session (allowing (semi-) automated processing). Concurrency shall also be ensured for accessing shared resources.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_016	
Name	Accessibility	
Category	Non-functional	
Description	Overall system / SoS shall be capable to access and process all the services and send/receive DDI in any operative condition and network condition during development and during runtime.	
Dependencies	-	
Conflicts	RQ_014	

Requirement	ID: RQ_017	
Name	Support for model-based dependability assessment techniques	
Category	Functional	
Description	Incorporate architectural and failure behaviour information in the DDIs. At a minimum, we would expect system or component architecture description to be included, as well as failure behaviour properties such as input/output/internal failure modes per input/output port of each system/component. Models shall be used for (semi-) automated	

	processing/analyses.
Dependencies	-
Conflicts	-

Requirement	ID: RQ_018	
Name	Represent Hazard/Risk Analysis artefacts	
Category	Functional	
Description	Information from Hazard/Risk Analysis artefacts shall be semi-automatically processed to enable dependability assessment and assurance. A description, classification of risk and identification of safety requirement per each hazard identified would be necessary. This information is important for high-level specification of hazards and associated high-level dependability requirements and (semi-) automated processing.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_019	
Name	Safety argumentation patterns	
Category	Functional	
Description	Incorporate the use of argument fragments or patterns, from which concrete assurance claims can be substantiated both during development and operation. This information is important for offline and online certification. Assurance arguments support certification of dependability properties. Argument patterns include dynamic elements within arguments, which can parameterize concrete arguments based on model and contextual information. The most common format for expressing assurance arguments is via structured argument graphs aka goal hierarchies. The Goal Structuring Notation (GSN) and Claims Arguments Evidence (CAE) notation both support specification of such hierarchies.	

Dependencies	RQ_012, RQ_013
Conflicts	-

Requirement	ID: RQ_020	
Name	Formal representation of dependability properties	
Category	Functional	
Description	Incorporating elements of formal specification to be captured within a DDI. To create executable DDIs and to allow run-time evaluation of system dependability properties. Specifically, incorporating formal specification elements would support runtime verification of dependability properties, which would then provide live evidence for the assurance of said properties. For example, state-based specification included within an element's DDI would allow the dynamic construction of Finite State Automata during runtime. The analysis of the automata would then verify that a dependability property is met (or has failed) and inform an assurance argument appropriately.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_021	
Name	Evolvability	
Category	Functional	
Description	DDI shall be meta-adaptive (scalable in case of data elements evolution). In such cases, the consistency of the unchanged data elements shall be ensured, and changes of the data elements incl. data attributes shall only be performed at the desired places.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_033	
Name	DDI evaluation result	
Category	Functional	

Description	Proper executable DDI evaluation algorithms shall be designed and implemented. The results shall conform to the SACM safety case and to the typical usage scenarios.
Dependencies	-
Conflicts	-

6.5 Runtime DDI

6.5.1 Requirements on the models needed to derive the runtime DDI

Requirement	ID: RQ_038	
Name	Scenarios and changes	
Category	Functional	
Description	Scenarios and their changes shall be described. Therefore, triggers, conditions, constraints, and assumptions of scenarios shall be described. Traceability between scenarios and changes shall be established.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_039	
Name	Safe behaviour	
Category	Functional	
Description	Safe behaviours (e.g. states with their transitions) shall be described.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_040	
Name	Failure behaviour	
Category	Functional	
Description	Failure behaviours (e.g. states with their transitions) shall be described.	
Dependencies	-	

Conflicts	-

Requirement	ID: RQ_041	
Name	Conditional requirements	
Category	Functional	
Description	Conditional requirements shall be defined and realized for the runtime DDI.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_042	
Name	Interface	
Category	Functional	
Description	The interfaces of the system under consideration shall be described.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_043	
Name	Configuration	
Category	Functional / non-functional	
Description	Configurations of the system under consideration at runtime shall be described.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_044	
Name	Dependencies of dependability goals	
Category	Functional and non-functional	
Description	Dependencies between dependability goals shall be defined.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_045	
Name	Transition between possible scenarios	
Category	Functional and non-functional	
Description	The possible transition between different scenarios shall be defined.	
Dependencies	-	
Conflicts	-	

6.5.2 Requirements on the execution of runtime DDIs

Requirement	ID: RQ_046	
Name	Access to runtime DDI models	
Category	Functional	
Description	Runtime DDI models shall be stored where the continuous access of the DDI runtime model is ensured.	
Dependencies	-	
Conflicts	-	

Requirement	ID: RQ_047	
Name	Changes of the fulfilment of the dependability goals	
Category	Functional	
Description	Possible changes of the fulfilment of the dependability goals during runtime shall be defined.	
Dependencies	-	
Conflicts	-	

7 Requirement Mapping

In this section, we describe the mapping of the requirements described in Section 6 to the DDI Engineering Stories and to the industrial use cases of the DEIS project.

7.1 DDI Engineering Stories – Requirements

This section describes the mapping of requirement of this innovation cycle to the current engineering stories. Table 2 provides an overview which of the requirements presented in Section 6 are relevant for which of the DDI Engineering Stories outlined in Section 4. The mapping shows that 15 of the requirements (25%) are relevant for all of the seven Engineering Stories which are the basis for the research conducted in the DEIS project.

Furthermore, the requirements have been prioritized according to their impact on the DEIS project. Priority 1 indicates mandatory requirements which can be possible obstacle for the DEIS project (50% of the requirements). Priority 2 requirements are "shall be addressed requirements" to ensure full potential of the technologies and applicability to the full spectrum of scenarios (30% of the requirements). The lowest priority (priority 3) requirements represents "nice to have" and not mandatory requirements which may only have limited effect on the DEIS project (20% of the requirements).

As a result, the following seven requirements are both relevant for all of the DDI Engineering Stories and mandatory for the DEIS project:

- RQ_001: Machine readable representation
- RQ_002: Composability
- RQ_002_01: Modularity
- RQ_026: Qualitative & quantitative data elements
- RQ_030: Data exchange
- RQ_036: Data portability identification (GDPR)
- RQ_037: Data perimeter (GDPR)

Requirements	ES1	ES2	ES3	ES4	ES5	ES6	ES7	Priority
RQ_001	x	x	x	x	x	x	x	1
RQ_002	x	x	x	x	x	x	x	1
RQ_002_01	x	x	x	x	x	x	x	1
RQ_002_02	x	x	x	x	x	x	x	2
RQ_003	x	x	x	x	x	x	x	2
RQ_004	x	x			x			1
RQ_005		x		x		x		3

RQ_006	x			x	x	x		2
RQ_007	x				x	x		1
RQ_008	x			x	x	x		2
RQ_022	x	x	x	x	x	x	x	3
RQ_023	x	x	x	x	x	x	x	3
RQ_024	x	x	x	x	x	x	x	3
RQ_034	x	x	x	x	x	x	x	2
RQ_035	x	x	x	x	x	x	x	2
RQ_036	x	x	x	x	x	x	x	1
RQ_037	x	x	x	x	x	x	x	1
RQ_009		x				x		1
RQ_009_01		x						2
RQ_009_02		x						2
RQ_009_04		x						3
RQ_009_05		x						3
RQ_009_06	x	x			x			3
RQ_009_07	x	x	x	x		x	x	2
RQ_009_08		x			x			3
RQ_009_09	x	x	x	x		x		3
RQ_009_10	x	x			x	x		2
RQ_009_11	x	x			x			1
RQ_010		x				x		2
RQ_011		x				x		1
RQ_025		x	x			x		1
RQ_026	x	x	x	x	x	x	x	1
RQ_027			x			x		1
RQ_028			x			x		1
RQ_012			x		x		x	2
RQ_013			x		x		x	2
RQ_014	x	x	x		x		x	2
RQ_029	x				x			2
RQ_030	x	x	x	x	x	x	x	1
RQ_031		x	x					1
RQ_032	x	x	x	x	x	x	x	3
RQ_009_03	x	x	x			x		3
RQ_015	x		x	x	x	x	x	1
RQ_016	x	x			x	x		2
RQ_017	x	x	x		x			1
RQ_018	x	x				x		1
RQ_019	x	x	x		x			2

RQ_020	x	x			x	x		1
RQ_021	x	x	x	x	x	x	x	3
RQ_033	x	x	x	x	x	x	x	2
RQ_038						x		1
RQ_039						x		1
RQ_040						x		1
RQ_041						x		1
RQ_042						x		1
RQ_043						x		1
RQ_044						x		1
RQ_045						x		1
RQ_046						x		1
RQ_047						x		1

Table 2: Mapping of requirements to DDI Engineering Stories and Prioritization of Requirements

7.2 Industrial Use Cases – Requirements

Since the industrial use cases as defined in D6.1 Use Case definition [9] are related directly to the different requirements DDI project requirements as depicted in Section 6. An overview of the mapping of the requirements to the industrial use cases is presented in the following table:

Requirements	Use Case 1 (automotive)	Use Case 2 (automotive)	Use Case 3 (railway)	Use Case 4 (healthcare)
RQ_001	x	x	x	x
RQ_002	x	x	x	x
RQ_002_01	x	x	x	x
RQ_002_02			x	
RQ_003		x	x	
RQ_004		x	x	x
RQ_005			x	
RQ_006	x	x		x
RQ_007	x		x	
RQ_008	x	x	x	
RQ_022	x		x	
RQ_023			x	
RQ_024			x	
RQ_034				
RQ_035	x			
RQ_036	x			x

RQ_037	x			x
RQ_009		x	x	
RQ_009_01		x	x	
RQ_009_02		x	x	
RQ_009_04		x	x	
RQ_009_05		x	x	
RQ_009_06	x	x	x	
RQ_009_07	x	x	x	
RQ_009_08	x	x		
RQ_009_09	x	x		
RQ_009_10	x	x	x	
RQ_009_11	x	x	x	
RQ_010	x		x	
RQ_011	x		x	
RQ_025	x		x	
RQ_026	x		x	
RQ_027			x	
RQ_028			x	
RQ_012	x	x		x
RQ_013	x	x		x
RQ_014	x			x
RQ_029			x	
RQ_030	x	x	x	x
RQ_031			x	
RQ_032			x	
RQ_009_03		x	x	
RQ_015	x	x		
RQ_016	x			
RQ_017			x	
RQ_018			x	
RQ_019			x	
RQ_020			x	
RQ_021			x	
RQ_033			x	
RQ_038		x		
RQ_039		x		
RQ_040		x		
RQ_041		x		
RQ_042		x		
RQ_043		x		

RQ_044		x		
RQ_045		x		
RQ_046		x		
RQ_047		x		

Table 3 : Mapping of the industrial use cases to the addressed requirements

8 Conclusion

This document provides an overview and summary on the requirements collected in the DEIS project. Therefore, this deliverable is summary of the requirements stated in the DEIS deliverables D2.1 [10], D2.2 [11], D2.3 [12]. Overall 60 requirements have been collected.

The requirements are organized around the following topics:

- General requirements
- Requirements regarding the DDI content
- Requirements regarding the exchange of DDIs
- Requirements regarding the DDI tooling
- Requirements regarding the runtime DDI

These requirements constitute the basis for the development carried out within the DEIS project. Therefore, we showed which of the requirements are relevant for which of the DDI engineering stories and for which of the industrial use cases of the DEIS project and introduced a prioritization.

9 Bibliography

[1] D. Schneider, M. Trapp, Y. Papadopoulos, E. Armengaud, M. Zeller and K. Höfig, "WAP: Digital dependability identities," *26th International Symposium on Software Reliability Engineering (ISSRE'15),* pp. 324-329, 2015.

[2] *Structured Assurance Case Metamodel (SACM), version 2.0,* Object Managment Group (OMG), 2016.

[3] A. Avizienis, J.-C. Laprie, B. Randell and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE transactions on dependable and secure computing,* pp. 11--33, 2004.

[4] DEIS Consortium, D3.1: Digital dependability Identities and the Open Dependability Exchange Meta-Model, 2018.

[5] F. Zappa, S. Tisa, A. Tosi and S. Cova, "Principles and features of single-photon avalanche diode arrays," *Sensros and Actuators A. Physical, Vol. 140, N. 1,* pp. 103-112, 2007.

[6] Shift2Rail, Shift2Rail Strategic Master Plan, Shift2Rail, 2015.

[7] CENELEC, *EN 50126 Railway Applications: The Specification and Demonstration of Dependability Reliability, Availability, Maintainability and Safety (RAMS),* 2002.

[8] CENELEC, *EN 50129 Railway Applications: Safety related electronic systems for signalling,* 2003.

[9] DEIS Consortium, "D6.1: Use Case Definition," 2017.

[10] DEIS Consortium, "D2.1: Project Requirements," 2017.

[11] DEIS Consortium, *D2.2: Refined project requirements for semi automation,* 2018.

[12] DEIS Consortium, *D2.3: Refined project requirements for full automation,* 2019.