Deliverable 1.9

Report on the Evolution of Co-Engineering Standards

Version 2.0

This project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737475. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Spain, France, United Kingdom, Austria, Italy, Czech Republic, Germany.

DISSEMINATION LEVEL		
X	**PU**	Public
	CO	Confidential, only for members of the consortium (including the Commission Services)

COVER AND CONTROL PAGE OF DOCUMENT	
Project Acronym:	AQUAS
Project Full Name:	Aggregated Quality Assurance in Systems
Grant Agreement No.:	737475
Programme	ICT-1: Cyber-Physical-Systems
Instrument:	Research & innovation action
Start date of project:	01.05.2017
Duration:	36 months
Deliverable No.:	D1.9
Document name:	Report on the Evolution of Co-Engineering Standards
Work Package	WP1
Associated Task	Task 1.2, 5b.2
Nature [1]	R
Dissemination Level [2]	PU
Version:	1.0
Actual Submission Date:	10-06-2020
Contractual Submission Date	31-10-2019
Editor: Institution: E-mail:	John Favaro, Silvia Mazzini Intecs

Change Control

Document History

Version	Date	Change History	Author(s)	Organisation(s)
0.1.0	5-08-2019	Initial Version derived from D1.3 with the same chapter structure	John Favaro, Erwin Schoitsch, Christoph Schmittner, Ricardo Ruiz, Jaime González Martínez, Ricardo Ruiz, Nicolas Ayache, Laurent Rioux, Silvia Mazzini, Marwa Gadala, Lorenzo Strigini, Daniel Kästner, Reinhold	INT, AIT, RGB, TAS, Clearsy, TRT, CITY, AbsInt, Siemens

[1] **R**=Report, **DEC**= Websites, patents filling, etc., **O**=Other

[2] **PU**=Public, **CO**=Confidential, only for members of the consortium (including the Commission Services)

			Heckmann, Martin Matschnig	
0.2.0	29-08-2019	Section 3.2: Gap analysis sections relative to the ESA Guide for Independent Software Verification and Validation and the ECSS-E-ST-40C, comparing with AQUAS approach for UC5 "Space multi-core architectures"	Olga Dedi, Robert Kaiser, Isaac Moreno, Abel Balbis Asenjo	HSRM, TASE
0.3.0	11-09-2019	Sections 2.1.2 UC2 Medicine and 2.5 Human Factors, Section 3.2: IP-XACT	Marwa Gadala, Emmanuel Vaumorin	City, MDS
0.4.0	09-10-2019	Section 3.2: Gap analysis of the VEL technical standard regarding its capability to enable design space exploration and trade-off analysis, Gap analysis of the ARINC653	Jabier Martínez, Vittoriano Muttillo, Ludovic Apvrille, Luigi Pomante Henrik Theiling	TEC, UAQ, MTTP SYSGO
0.5	15/10/2019	Section 2.2: MISRA C Section 3.2: Arcadia Section 2.1: updates for Unmanned Aerial Systems (UAS) and for UC3 Railways, Section 2.3: Update on Automotive standards, Smart Manufacturing, Robotics, Adaptive Open Systems, IoT, AI Section 3.2: IEC 61508 and IEC 63187 – Safety and Cybersecurity, current status of discussions, Section 4.2 updates on AIT activities (O9: Cross-domain requests and O10: Promoting awareness) Section 2.1.1: UC1 ATM updates Sections 2.2.1 and 4.2 updates on AMT activities Section 3.2: Gap analysis of ANSI/ISA-62443-3-3 (99.03.03)-2013 Section 3.2: Gap analysis of ISO/IEEE11073-00103 Section 2.2: FMI	Reinhold Heckmann Charles Robinson, Matthieu Pfeiffer, Jabier Martínez, Shuai Li Erwin Schoitsch Jose Cordero Mario Winkler Petr Mlynek, Radek Fujdiak, Pavel Mrnustik Tomas Vojnar, Maros Barabas, Bohuslav Krena Pacome Magnin	Absint TRT, All4Tec, TEC, MAG, CEA AIT Integrasys AMT Trustport BUT Trustport BUT SISW

		Executive Summary revision, Restructuring and renaming of Section 2.3 Requirements for MARTE evolution Revision of Section 4.1 Overall approaches and Section 4.2 Current approaches Chapter 5 revision to update reporting of other activities, Creation of Chapter 6 specific for conclusion, and Annex A to for SE Goals KPI reporting	Silvia Mazzini	INT
0.6	21/10/2019	First version for internal review	Silvia Mazzini	INT
1.0	31/10/2019	Final version delivered at M30	Silvia Mazzini	INT
2.0	10/06/2020	Second version for delivery at M38 Section 3.2.3: updates on ARCADIA gap analysis Section 4.2: updates including all the activities performed at M38 Update of conclusions Annex A: update of Key Performance Indicators	Silvia Mazzini Charles Robinson, Matthieu Pfeiffer, Jabier Martínez, Shuai Li Marwa Gadala, Lorenzo Strigini Reinhold Heckmann Erwin Schoitsch Petr Mlynek, Radek Fujdiak, Pavel Mrnustik Tomas Vojnar, Maros Barabas, Bohuslav Krena, Ludovic Apvrille ken Sharman Silvia Mazzini	INT TRT, All4Tec, TEC, MAG, CEA CITY Absint AIT Trustport BUT Trustport BUT MTTP ITI INT

Distribution List

Date	Issue	Group
	Document alignment	application.domain.leads@aquas-project.eu
	Final Version	application.domain.leads@aquas-project.eu, leaders@aquas-project.eu

TABLE OF CONTENTS

TABLE OF FIGURES

Table OF TABLES

Executive Summary

D1.9 is the final report on standard evolution activities performed in WP1 (Task 1.2 Technical Coordination on Standard Evolution) and WP5 (Task 5b.2: Standardisation) and is an update of the AQUAS Deliverable D1.3. The document keep the same chapter structure of D1.3, major changes are documented in the Document History, while minor updates are not in evidence.

Regular submission of version 1.0 of this this deliverable occurred at M30. The current version 2.0 update extends the reporting on the final progress at M38.

The interplay between dependability attributes such as safety / security / performance is being increasingly accepted by all involved stakeholders and discussions on how to react to this development in standardization is ongoing. Related standards in multiple domains are currently under revision or (especially in the case of security standards) for the first time under development. For IoT and the increasingly open and dynamic systems, it will be necessary to regulate and consider multiple dependability attributes. Due to the continuing involvement of AQUAS partners in standardization activities, AQUAS has developed many opportunities to influence standardization. The AQUAS work on the evolution of co-engineering standards takes as its point of departure a body of existing work that is relevant to the project in various ways:

- Existing standards that directly govern the development of the demonstrators in the AQUAS use cases;
- Transversal standards that are not specifically relevant to the selected use cases in AQUAS, but which are relevant to AQUAS objectives for tool interoperability, modelling, code development, etc.;
- Ongoing standardisation initiatives which might not necessarily have produced results yet, but can be a source of guidance for AQUAS.
- Human Factors standards that have essential roles in safety, security and system performance.

In multiple domains which already have safety as an established property, security is becoming a new issue. The introduction of performance into co-engineering is extremely recent, and few standards are treating it to date. Thus, we must rely currently on experience with cybersecurity and safety dimensions in the standards developing organizations to understand how they are currently addressing the topic of co-engineering. The establishment of a unified risk assessment / management regime in co-engineering for all three dimensions remains a significant challenge. Furthermore, each of the three co-engineering dimensions treats “best practices” in development in different ways, making it difficult to harmonize the standardisation of development according to each of the three dimensions. This also remains a challenge for co-engineering standards.

The AQUAS consortium was well aware that it will not always be possible to synchronize its standards-influencing efforts with the windows of opportunity that will arise during the evolutionary cycles of the standardization groups. Nevertheless, activities were engaged that are useful even in the absence of perfect synchronisation with the standards renewal cycles.

Significant activities reported in this document are the analyses of co-engineering gaps in current standards. Representative standards were addressed, as examples of issues in co-engineering, and more extensively standards that are actually influenced for evolution, or candidates.

The following approaches to influence standards are considered:

- Reports and change request packages valid for future revision cycles,
- Presentations to standards committees and working groups,
- Guidelines for the usage and interpretation of standards in particular ways,

- Dissemination of Gap Analysis.

In order to channel upcoming standards evolution activities in the project into the most effective paths, the project put in place mechanisms for tracking responsibilities, on-going efforts and progress toward the achievement of the objectives.

The strategies outlined were employed regardless of standards revision cycles and ensured that the project was able to produce a set of change requests, influence standards and raise awareness beyond the lifetime of the project. A large number of actionable requirements for standards evolution have already been submitted at M30.

1 Introduction: Current Challenges in Standards Evolution

The interplay between dependability attributes such as safety / security / performance is being increasingly accepted by all involved stakeholders and discussions on how to react to this development in standardization is ongoing. Related standards in multiple domains are currently under revision or (especially for security standards) for the first time under development. For IoT and the increasingly open and dynamic systems, it will be necessary to regulate and consider multiple dependability attributes. Due to the ongoing involvement of AQUAS partners in standardization activities, AQUAS has a window of opportunity to influence standardisation.

However, it is still difficult to address such issues in a cross-domain way. Different domains have established safety standards, and security standards are partially designed to interact and extend existing standards. Therefore, we do not expect much overlap between the domains in standardization. A positive counterexample is the acceptance of IEC 62443 [10] as a template for future cybersecurity standards for additional domains like railways.

Another challenge is related to the current lack of focus on **performance** considerations in standardisation activity, in contrast to the increasing interest in safety / security co-engineering. This issue is finally being given a push by the emergence of autonomous applications in multiple domains, and standardization groups are beginning to realize the relevance and importance of the performance dimension. But the challenge is significant and open-ended, representing at the same time a potential opportunity for AQUAS.

Besides multi-concern standardization, tool interoperability will also play an important role in the success of the AQUAS activities. Only accepted and well-specified interoperability standards will allow the seamless interoperability between AQUAS internal and external tools and support the automation of co-engineering processes.

Each of the above-described challenges are addressed in more detail in the sections that follow.

The AQUAS project has set the following objectives (the numbers correspond to those mentioned in the Description of Work):

- **Objective 9**: Contribute to the improvement of standards to address co-engineering, by submission of change requests to at least 1 standard for each of the AQUAS use case domains.
- **Objective 10**: To promote awareness and bring results of AQUAS into at least two international standards in the functional safety and security area with respect to safety, security and performance co-engineering.
- **Objective 11**: To influence actively two international standardization groups focused on frameworks for the coordination of safety, security and reliability of automation.
- **Objective 12**: To promote awareness and bring results of AQUAS into at least two other international engineering standards, such as OMG, or FMI.

In the following sections, a basis for the AQUAS approach to meet these objectives are identified.

2 Standards influencing AQUAS

The AQUAS work on the evolution of co-engineering standards takes as its point of departure a foundation of existing work that is relevant to the project in various ways:

- Existing standards that directly govern the development of the demonstrators in the AQUAS use cases. These represent “bedrock” in the project: real standards that have a concrete effect on real application development;
- Existing standards that are not specifically relevant to the selected use cases in AQUAS (for example, they might be in different domains), but which have already addressed aspects of co-engineering that are relevant to AQUAS objectives and which could help to achieve those objectives;
- Standardisation initiatives that are currently ongoing but have not necessarily produced results yet. These can provide valuable guidance in how to proceed in our own work.

The following sections further elaborate on these points.

2.1 Standards and the AQUAS Use Cases

2.1.1 UC1 Air Traffic Management

The ATM use case offers one of the first opportunities for the AQUAS project to break potential new ground in the examination of **performance** as a first-class citizen in the treatment of co-engineering in standards. As reported earlier [4], the so-called SWIM profiles [1][2][3] developed within the SESAR projects are the primary sources of requirements on performance and security. But these profiles were originally developed in the context of commercial aviation, long before the advent of remote-controlled, unmanned vehicles (drones), and therefore placed their principal emphasis on human safety. With the increased use of telecommunications, and the attendant need for ultra-reliable low-latency communications and enhanced broadband communications capacity, **performance** has become an integral factor in requirements engineering for this category of application. The mission critical nature of both military and civilian drone applications, combined with the reliance on wireless communications, has also promoted security aspects to first-class citizenship. Now true co-engineering of safety, performance, and security is needed for this class of application.

Therefore, the AQUAS analyses of the trade-offs between safety, performance, and security in the ATM use case will provide a great opportunity to integrate co-engineering experience of AQUAS for UAVs into current initiatives of regulation and standardisation.

The regulation and standardization in UAV domain are currently at an evolutionary stage in order to decrease potential threats to public safety and security and integration with current ATM systems. The applied security and performance requirements in this use case have been based on the next standards or guides:

- **EUROCONTROL (SWIM Profiles):** These specifications contain requirements for system interfaces and for IT infrastructure capabilities required to enable a reliable, secure and efficient exchange of information.
- **International Electrotechnical Commission (IEC 62443 - Security for Industrial Process Measurement and Control: Network and System Security):** This standard describes security requirements and controls which are essential for building a security framework in a system as they explicitly define security measures which must be present in the system in order to assure its protection.

For AQUAS, we highlight the most interesting working groups currently in operation in terms of regulation, guidelines and standardization:

- CEN-CENELEC (D5/WG8): The Standardization efforts are related principally to classification, design, manufacture, operation (including maintenance) and safety management of UAS (Unmanned Aircraft Systems) operations. These standard applies to the commercial and recreational use of unmanned aircraft systems and thus to consumer products and technical work equipment.
- EASA (Expert Group on Drones - E03533): This working group has the aim to facilitate the integration of drones in very low-level airspace (i.e. below 150 meters) and preserve the high level of safety in the entire European airspace. It has the following mission:
    - to act as a sounding board for the conception and implementation of the EU drone policy.
    - to advise and assist the Commission with the implementation of actions that can foster and accelerate the integration of drones in the aviation system and the emergence of a suitable operational environment and infrastructure for drones flying at low altitude, including over urban areas.
    - to build upon the operational infrastructure and services to foster the development of a drone services market and of a robust, dynamic and innovation-oriented supporting framework.
    - to draw on best practice and “lessons learned” in other regions and/or in other industrial and service sectors that can be instrumental for the purposes referred to herein.
- EUROCAE (WG-105)**:** The aim is to develop standards and guidance documents that will allow the safe operation of UAS in all types of airspace, at all times and for all types of operations. Specially, within that working group we highlight the next subgroups:
    - SG23 - Security.
    - SG33 - UTM Geo-Fencing.
    - SG62 - GNSS for UAS.
- JARUS: This is a group of experts from the National Aviation Authorities (NAAs) and regional aviation safety organizations. Its purpose is to recommend a single set of technical, safety and operational requirements for the certification and safe integration of Unmanned Aircraft Systems (UAS) into airspace and at aerodromes. The objective of JARUS is to provide guidance material aiming to facilitate each authority to write their own requirements and to avoid duplicate efforts. Within the different working groups we highlight the following:
    - WG5 - Command, Control & Communications
    - WG6 - Safety & Risk Management
    - WG7 - Concept of Operations
- **ISO TC20/SC16 (TC20 – Aircraft and Space Vehicles, SC16 Unmanned Aircraft Systems (UAS))** recently started standardization in the field of UAS including, but not limited to, classification, design, manufacture, operation (including maintenance) and safety management of UAS operations (UTM – UAS Traffic Management). This committee has at the moment only standards under development which means we have an ‘Open Window of Opportunity’:

- ISO/CD 21384-1 [Under development] Unmanned aircraft systems -- Part 1: General specification
- ISO/CD 21384-2 [Under development] Unmanned aircraft systems -- Part 2: Product systems
- ISO/CD 21384-3 [Under development] Unmanned aircraft systems -- Part 3: Operational procedures
- ISO/CD 21895 [Under development] Categorization and classification of civil unmanned aircraft systems
- ISO/AWI TR 23629-1 [Under development] UAS Traffic Management (UTM) -- Part 1: General requirements for UTM -- Survey results on UTM

Other organisations have developed particular standards for their field of application, e.g. the 'National Fire Protection Association' (US): 'NFPA 2400: Standard for Small Unmanned Aircraft Systems (sUAS) used for Public Safety Operations'. Additionally, the relevant standards for Air Traffic Management (ATM), which are mainly concerned about the potential negative impact of flying drones on civil aviation, have to be taken into account, particularly if larger drones (transporting goods and persons) are involved.

2.1.2 UC2 Medicine

This section updates the similar section in Deliverable 2.2.2 [5] by adding some additional standards that are being considered; updates are in **boldface**.

Medical devices are strongly regulated in the European Union. The current medical devices directive 93/42/EEC is still applicable, but a new medical device regulation 2017/745 was approved in 2017 and will be fully applicable in 2020. These regulations specify the requirements that a medical device legally placed on the European market must satisfy, and state that a sufficient condition for satisfying requirements is compliance "with the relevant national standards adopted pursuant to the harmonized standards", which makes such compliance with the latter the natural path for industry to follow. Standards currently harmonised with the medical devices directive 93/42/EEC can be found at [9].

The main standard considered in the medical use case is EN 60601-1:2006, titled, "Medical Electrical Equipment. General requirements for basic safety and essential performance". This standard is a general safety standard applicable to any type of medical device. It has several associated collateral standards that are mandatory when applicable. In this use case, the relevant ones (which are technically equivalent to the international IEC 60601-standards series) are:

- EN 60601-1-2: 2007. "Collateral standard: Electromagnetic compatibility"
- EN 60601-1-6: 2010. "Collateral standard: Usability"
- EN 60601-1-8: 2007. "Collateral standard: General requirements, tests and guidance for alarm systems in medical electrical equipment and medical electrical systems"
    - **For research purposes, we are also considering IEC 60601-1-8/AMD2 ED2: Amendment 2 – Medical electrical equipment – Part 1-8: General requirements for basic safety and essential performance – Collateral Standard: General requirements, tests and guidance for alarm systems in medical electrical equipment and medical electrical systems. This is a draft and is not yet to be regarded as a standard, but provides insight into the direction of changes proposed.**
- EN 60601-1-10: 2008. "Collateral standard: Requirements for the development of physiologic closed-loop controllers"

There are also particular standards related in the standard series, that are related to the safety of specific types of medical devices. The standards applicable to this use case are:

- EN 60601-2-10:2015. "Particular requirements for the basic safety and essential performance of nerve and muscle stimulators". This standard specifies particular requirements related to the measurement of neuromuscular transmission.
- EN 80601-2-30: 2010. "Particular requirements for the basic safety and essential performance of automated non-invasive sphygmomanometers". This standard specifies particular requirements related to the non-invasive measurement of blood pressure.

Although the IEC 60601 series also covers aspects such as product life cycle, there are specific standards that directly address such aspects in more detail:

- EN 62304:2006. "Medical device software - Software life cycle processes". The EN 62304 standard requires following the well-known V-model for the software life cycle processes of a medical device, but the rest of the standards normally include specific requirements not related with the product life cycle.
- **ISO 13485:2016. "Medical Devices - Quality Management Systems - Requirements for Regulatory Purposes"**
- **IEC 61508. "Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems" covering aspects to be considered when electrical/electronic/programmable electronic systems are used to carry out safety functions"**

In AQUAS we have paid special attention to the effects of usability standards on safety, performance and security:

- EN 62366-1:2015. "Medical devices - Application of usability engineering to medical devices" and as guidance to this standard: **IEC/TR 62366-2:2016. "Guidance on the Application of Usability Engineering to Medical Devices".**

For issues related to symbols and labelling of medical devices, the following standards are also considered in this use case:

- **ISO 15223-1:2016. "Medical Devices – Symbols to be used with medical device labels, labelling and information to be supplied".**
- **IEC 60878:2015. "Graphical symbols for electrical equipment in medical practice"**

Some of the aforementioned standards have issued several amendments since their approval and are periodically revised.

According to medical device regulation in the European Union, it is mandatory to perform a risk assessment starting with the initial design of a medical device and during all phases of its lifecycle. This risk assessment must be performed in compliance with the following standard:

- EN ISO 14971:2012. "Medical device - Application of risk management to medical devices"

There are **no harmonised standards related to cybersecurity**, but some security-related standards that should be considered are:

- ISO 27799:2016. "Health informatics - Information security management in health using ISO/IEC 27002". It is a guideline for the application of the ISO/IEC 27002 standard to health informatics. ISO 27799 and ISO/IEC 27002 taken together define what is required in terms of information security in healthcare to maintain the confidentiality, integrity and availability of personal health information.
- ISO/IEEE 11073. "Health informatics - Medical health device communication". This set of standards enables communication between medical, health care and wellness devices with external computer systems.
- HL7 Standards. This family of standards is related to clinical information exchange and is widely used for the communication between medical devices and Patient Data Management Systems (PDMS) in hospital environments.

2.1.3 UC3 Railway

The basic safety-related standards for railways are EN 50126, EN 50128, EN 50129 and EN 50159 (See [6]).

- EN 50126 – Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS) – Part 1: Basic Requirements and generic process.
- EN 50128 - Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems.
- EN 50129 - Railway applications – Communications, signalling and processing systems – Safety related electronic systems for signalling.
- EN 50159 - Railway applications - Communication, signalling and processing systems. Safety-related communication in transmission systems.

CEN TS 50701 (Railway Applications – Cybersecurity)

In all the previously mentioned railway standards "security" is not mentioned except in the context of physical access, based on the traditional isolation of railway signalling and communication systems from regular public systems. With increased use of public facilities and wireless communication and control systems, e.g. the European Train Control System, the "security-aware safety" considerations in standardization are now starting also in the railway sector (see Figure 1). CyberSecurity is a relatively new topic which has become very important, not only for railways, but for all Critical Infrastructures. Due to this CENELEC decided to work on a railway specific adaptation and interpretation of the (partially still emerging) IACS CyberSecurity Standard IEC 62443.

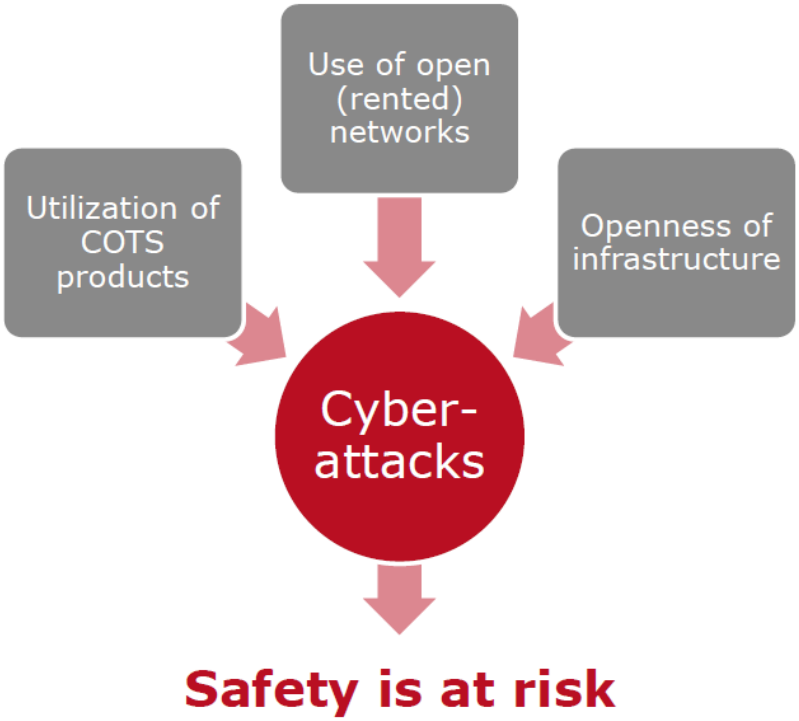

Figure 1: Railway systems – from "proprietary" to "openness" – new risks (Source: DB Netz AG, RSSRail 2019)

DKE, in Germany, is integrating requirements from IEC 62443 in the railway standards (proposal, addressing EN 50129 and EN 50159 issues) by DIN VDE V 0831-104 "Electric signalling systems for

railways – Part 104: IT Security Guideline based on IEC 62443". Work was transferred to CENELEC TC9X. SC9XA SGA 16 was asked to prepare a report for SC9XA setting out the proposed scope of a future standard on Cybersecurity. The result were the following recommendations:

- To create an EN (Standard, not only report (TR) or Specification (TS)
- To set the scope for a New Work Item (NWI) proposal
- To embed this EN in an overall framework within TC9X

In the end, it was decided to create a TS, which tackles the cybersecurity issue not only from the signalling safety and communication viewpoint, but also from the top-level viewpoint (see Figure 2) for the whole "railway system".

The CENELEC TS 50701 was released as a draft in mid of 2019. With TS 50701 a Cybersecurity standard will become available which covers not only Signalling, Rolling Stock or Fixed Installations but the whole Railway System. After release of the TS a good tool for fulfilling the NIS directive as well as the national transpositions will be available, due to the participation of ERA and ENISA to the working group.
In this workshop experts of the responsible working group for TS 50701 will present and discuss the key aspects like life cycle, system definition, risk analysis, security requirements as well as operation and maintenance requirements.

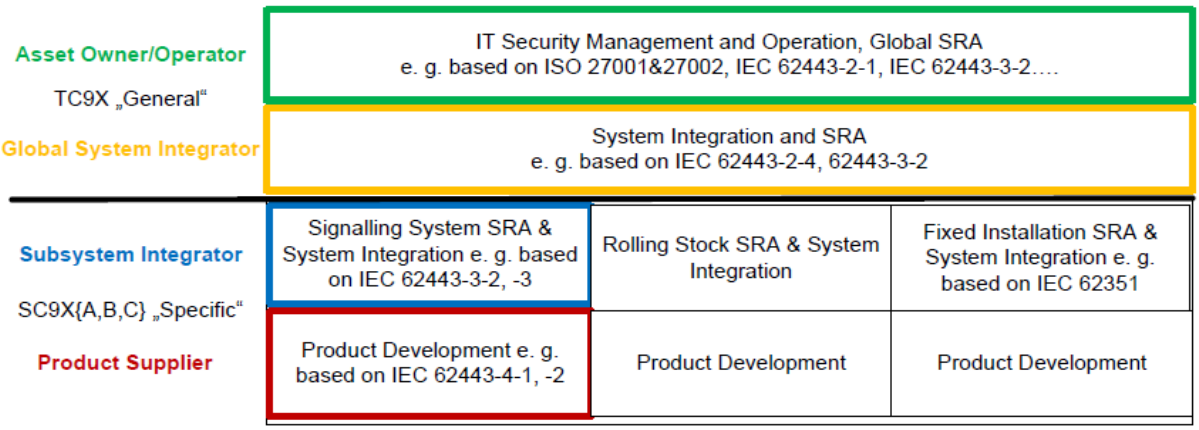

Figure 2: Considering Cybersecurity in Railway Standards

An example of an adaptation of other railway standards, e.g. for rolling stock, to the functional safety standards as already common in signalling, may be found in the new EN 50657 "Railways Applications – Rolling stock applications – Software on Board Rolling Stock". After a withdrawal of the old version, a new one has been issued (1 December 2017). It replaces EN 50128 for rolling stock and takes over most concepts from that standard. It is a bit less strict with respect to low safety integrity, for instance lower documentation requirements, and in this sense SIL0 has been renamed to "Basic integrity".

Based upon discussions with experts, it appears that the aforementioned standards cover development processes regarding safety and security in a rather open way: the objectives are described, but there are no constraints on the means. Achieving the required level of safety or security entirely depends on the safety or security demonstrations. For instance, if one is using tests in a part of the project, they will have to provide a demonstration as to why they satisfy the considered requirements. But the same approach would apply if another technique, say formal methods, was used.

2.1.4 UC4 Industrial Drive

As stated in earlier deliverables, the governing **safety-related** standard for this use case is IEC 61508 – "Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems". [11]

With respect to functional safety, the standard "Functional safety for adjustable speed electrical power drive systems" (IEC 61800-5-2) defines safety functions performing monitoring and/or safety-relevant controlling tasks, e.g. Safe torque off (STO). It was discussed previously [7] that, for example, this standard contains requirements (such as uncontrolled communication during an emergency situation) that could create safety and security interference.

However, the newer developments concern **cybersecurity standardisation** in this sector. As reported earlier in the AQUAS project, IEC 61508 is now in a relatively stable phase, having recently undergone a significant revision (Second Edition in 2010), in which some aspects of cybersecurity are touched upon, particularly in the section on Hazard Analysis. Currently ongoing maintenance activities for 61508 are reported and discussed under section 3.2.9 . The new development to report is the evolution of the set of IEC 62443 standards.

IEC 62443 "Industrial communication networks -Security for industrial automation and control systems" is a series of standards and technical reports with the goal of improving safety, availability, integrity, and confidentiality of IACS (Industrial Automation and Control Systems). Figure 3 gives an overview on the planned set of standards and technical reports.

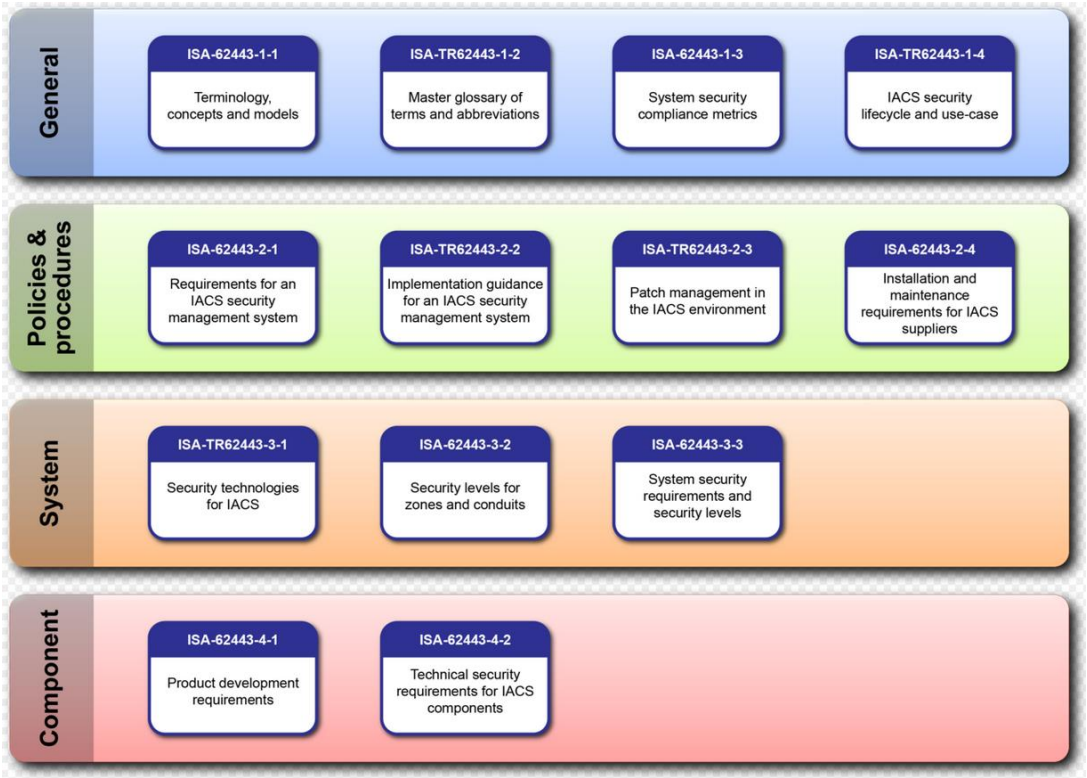

Figure 3: Planned parts of IEC 62443 (*source: IEC*)

The standards define procedures for implementing electronically secure IACS. Their guidance applies to end-users (i.e. asset owner), system integrators, security practitioners, and control systems manufacturers responsible for manufacturing, designing, implementing, or managing IACS.

In these standards, the main concepts relevant for the AQUAS approach and in particular for the Industrial Drive use case are:

- The zone-conduit concept described and the respective workflow in IEC 62443-3-2 and IEC 62443-3-3, which allows a structured cybersecurity risk analysis with targeted measures for the safety-critical zones, and
- The concept of security levels (SL): SL-T (target SL), SL-C (SL capabilities), and SL-A (achieved SL) introduced also in IEC 62443-3-2.

- EC 62443-4-2 Security for industrial automation and control systems - Part 4-2: Technical security requirements for IACS components, was issued 2019. It provides detailed technical control system component requirements (CRs) associated with the seven foundational requirements (FRs) described in IEC TS 62443-1-1 including definition of the requirements for control system capability security levels and their components, SL-C(component).

The IEC 62443 series of standards is intended to be used across industrial control segments and has been approved by many countries. The concepts have also influenced the railway domain and have commonalities with the AQUAS approach.

2.1.5 UC5 Space Multicore

The European Cooperation for Space Standardization (ECSS) represents a cooperative effort of the European Space Agency (ESA), national space agencies and European industry associations for the development of a coherent, single set of consistent space standards for use by the entire European Space Community. The objective of creating this organization was to produce standards to be used throughout the European space business. Therefore, the European Space Agency (ESA) contractors must adhere to the standards created by this organization [8].

The result of this effort is the ECSS series of Standards (ST), Handbooks (HB) and Technical Memoranda (TM) organized in four branches:

- M: Management Standards
- Q: Product Assurance Standards
- E: Engineering Standards
- U: Usability Standards

Among them, the most relevant standards for the AQUAS project are:

- ECSS-E-ST-10 “Space Engineering” specifies the system engineering implementation requirements for space systems and space products development. Specific objectives of this standard are: to implement the system engineering requirements to ensure a firm technical basis and to minimize technical risk and cost, to specify the essential system engineering tasks, their objectives and output, to implement integration and control of engineering disciplines and lower level system engineering work, to implement the “customer-system-supplier model” through the development of systems and products for space applications.
- ECSS-Q-ST-30 “Space product assurance (dependability)” defines the requirements for a dependability assurance programme in space projects. This standard calls for the use of dependability analysis techniques, tailored to match the generic requirements in each project, to address the hardware, software and human functions composing the system.
- ECSS-Q-ST-40 “Space product assurance (safety)” defines the safety programme and the technical safety requirements for space projects.
- ECSS-E-ST-40 “Software” focuses on space software engineering process requirements and their expected outputs, putting a special emphasis on the system-software relationship and on the verification and validation of software items.
- ECSS-Q-ST-80 “Space product assurance – Software product assurance” defines a set of software product assurance requirements to be used for the development and maintenance of software for space systems. The objective is to provide adequate confidence to the customer and to the supplier that a software (developed or reused) satisfies its requirements throughout the system lifetime. In particular, that the software performs properly and safely

in its operational environment and meets the quality objectives agreed for the project. The requirements defined in the ECSS-Q-ST-80 standard deal with quality management, process definition and quality characteristics of software products during the whole project life cycle.

A gap analysis of these standards is presented in a subsequent section (Section 3.2.1).

2.2 Transversal standards activity influencing AQUAS

There is current standardisation activity that is not immediately associated with a particular domain, and therefore is analysed separately. Primarily this concerns development standards, modelling standards, standards adopted by tools, such as tool interoperability standards, in relationship with tools being employed in AQUAS.

Other transversal standards are discussed in Section 3.2 for gap analysis.

2.2.1 OMG standards activity

The OMG (Object Management Group) is an international, open membership, not-for-profit organization for the development of technology standards. OMG standards are driven by vendors, end-users, academic institutions, and government agencies. OMG Task Forces develop enterprise integration standards for a wide range of technologies.

The OMG technology adoption process [24] is quite elaborate: it starts with a Request for Proposals (RFP). An RFP is a statement of industry need and an invitation to the software supplier community to provide a solution, based upon requirements stated within. The process of identifying need is a culmination of experience within an OMG technical group (be it a Task Force, a Special Interest Group or a Subcommittee) and solicitation of industry recommendation. Any Contributing, Domain or Platform Member of the OMG in good standing may propose specifications for adoption by OMG in response to an RFP. The initial submissions in response to an RFP are developed and presented to the sponsoring Task Force, which provides feedback to the submitter(s), a determination is made for the need for revised submissions. Revised submissions can be iterated more times until the approval by the OMG Membership is reached, and a further finalization step is issued.

Several standards and initiatives at OMG are related to AQUAS activities. The OMG “modelling standards”, such as the Unified Modelling Language (UML), and in particular the Systems Modelling Language (SysML) and MARTE (Modelling and Analysis of Real-time and Embedded systems), are at the very heart of AQUAS tool interoperability activities.

SysML is a general-purpose modelling language for systems engineering applications. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems.

MARTE is a UML profile defining foundations for model-based description of real time and embedded systems. These core concepts are then refined for both modelling and analysing concerns. Modelling parts provides support required from specification to detailed design of real-time and embedded characteristics of systems. In addition, facilities to annotate models with information required to perform specific analysis are provided. MARTE focuses especially on performance and schedulability analysis. It is a general framework for quantitative analysis which may be specialized for other kind of analysis.

They constitute the formal basis for specifying the artefacts to be modelled, annotated for analysis and exchanged among many of the AQUAS tools, and are being followed closely by some AQUAS partners.

In addition they are being extended in AQUAS to support modelling and co-engineering of safety, security and performance (e.g. by Intecs for the CHESS tool extensions), and influenced for their

evolution, as in the case of MARTE by TRT, CEA and Intecs, or, in the case of SysML, by AMT (see **Errore. L'origine riferimento non è stata trovata.**)

2.2.2 OSLC

In the context of software and system interoperability and integration, the Open Services for Lifecycle Collaboration (OSLC) initiative is a joint effort between academia and industry to improve data sharing and interoperability among applications by applying the Linked Data principles: "*1) Use URIs as names for things. 2) Use HTTP URIs so that people can look up those names. 3) When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL) and 4) Include links to other URIs, so that they can discover more things*". See Figure 4.

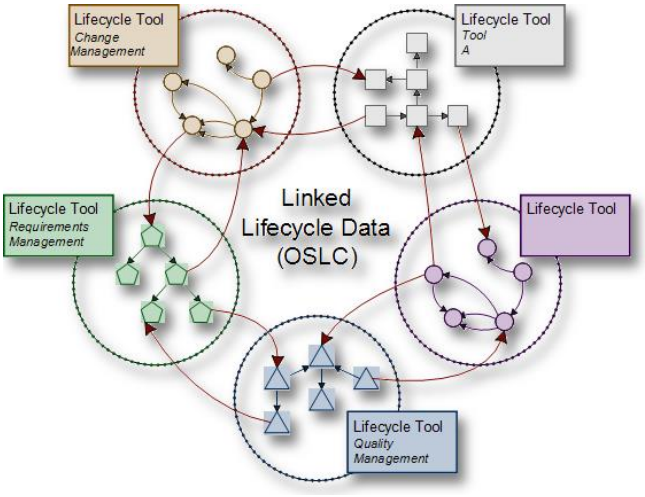

Figure 4: OSLC Concept of Linked Lifecycle Data

Led by the OASIS OSLC working group[3], OSLC is heavily based on web standards (as can be seen in the description above), in particular RDF for providing a common *data model*, and HTTP for providing a common *protocol*.

The objective, particularly relevant to AQUAS, is to provide Product Lifecycle Management tools with agreement on how and which data to share. One issue is that the artefacts generated during the AQUAS lifecycle are not necessarily already defined, in part because of the CPS nature of AQUAS relevant systems. For example, simulation models or physical circuits are examples of potential artefacts whose OSLC "resource shape" is not yet defined. These may have to be defined during the AQUAS project, depending on their appearance in the use cases.

Essentially, by taking advantage of the Linked Data principles and Web standards and protocols, the OSLC effort is attempting to create a family of web-based specifications for products, services and tools that support all the phases of the software lifecycle. A number of industry platforms such as PTC Integrity[4], Siemens Team Center[5], IBM Jazz Platform[6] or HP PLM[7] are now offering OSLC interfaces for different types of artefacts.

Note, however, that data exchange does not necessarily imply integration. From service providers to data items, an integration strategy is required to represent, store, search and coordinate collaboration

[3] http://www.oasis-oslc.org/

[4] http://www.ptc.com/application-lifecycle-management/integrity

[5] http://www.plm.automation.siemens.com/en_us/products/teamcenter/

[6] https://jazz.net/

[7] http://www8.hp.com/us/en/business-services/it-services.html?compURI=1830395

between software artefacts metadata and contents. In this light, the OSLC initiative is currently following this approach, and is having an impact on the main players in the software and systems industry. Nevertheless, it only covers a restricted type of artefacts and some crosscutting and basic services for reuse, such as indexing or retrieval, must be provided by all third parties. Within AQUAS, these restrictions will be taken into consideration with regard to the effort that would be required to fill the existing gaps in the OSLC conceptual and practical offerings, in order to decide on the relative costs and benefits of an OSLC oriented solution to interoperability of AQUAS tools.

It should be mentioned that the outcomes of the Horizon 2020 Support Action CP-SETIS on tool interoperability, which are based on the OSLC approach and have been extended to further interoperability standards and guidelines as a multi-standards platform, is supported by ARTEMIS-IA as a small community project (the IOS-ICF, *Interoperability Coordination Forum*). AQUAS partner AIT was a partner in CP-SETIS driving the standardization agenda for CPS, and is further involved in ICF.

Note also that AQUAS Deliverable D4.2 (November 2018) describes the usage of OSLC in AQUAS for interfacing ITI's A2K tool with BUT's ANaConDA, and explains the rationale for choosing OSLC for that purpose.

2.2.3 INCOSE

The International Council on Systems Engineering (INCOSE) is a not-for-profit membership organization founded to develop and disseminate the interdisciplinary principles and practices that enable the realization of successful systems. INCOSE is focused on producing state-of-the-art work products that support and enhance the Systems Engineering discipline’s visibility in the world.

INCOSE promotes a Systems Engineering Vision [25] to inspire and guide the direction of systems engineering across diverse stakeholder communities, which include different engineering disciplines and tool vendors and prepare the System Engineering Handbook [26] to describe key processes and activities performed in systems engineering.

INCOSE, together with the Institute of Electrical and Electronics Engineers Computer Society (IEEE-CS), and the Systems Engineering Research Center (SERC) provides also the funding and resources needed to sustain and evolve the Guide to the Systems Engineering Body of Knowledge (SEBoK) [53] and make it available as a free and open resource to all. The SEBoK provides a compendium of the key knowledge sources and references of Systems Engineering organized and explained to assist a wide variety of users.

Terms and knowledge from INCOSE and the SEBoK have been considered as a reference for the AQUAS project.

2.2.4 MISRA C/C++/AUTOSAR

Over the years, *The Motor Industry Software Reliability Association* (MISRA Consortium) published a series of software development guidelines for the programming language C and C++. Originally, these guidelines sought to promote the safest possible use of these languages by developers of automotive embedded systems. Today, MISRA C has evolved to a widely accepted model for best practices by leading developers in sectors including automotive, aerospace, telecom, medical devices, defense, and railway.

The initial *MISRA-C:1998 Guidelines for the Use of the C Language in Vehicle Based Software* published in 1998 were later revised to the *MISRA-C:2004 Guidelines for the use of C language in critical systems*, pertaining to the C language defined by ISO/IEC 9899:1990 standard amended and corrected by ISO/IEC 9899/COR1:1995, ISO/IEC 9899/AMD1:1995, and ISO/IEC 9899/COR2:1996.

The *MISRA C++:2008* ruleset defines a subset of C++ suitable for use in safety-critical systems using techniques similar to those within MISRA C. It gathers existing C++ guidelines from many diverse

sources into a single repository, but also adds new guidelines so as to enhance the state-of-the-art. MISRA C++:2008 aims to establish a single, generic set of guidelines for the use of C++ in safety-critical systems that are understandable to the majority of programmers.

The *MISRA C:2012 Guidelines for the use of the C language in critical systems*, like their predecessors, define a subset of C in which the opportunity to make mistakes is either removed or at least reduced. While previous editions of MISRA C had been based on the 1990 ISO definition of C, the MISRA C:2012 guidelines are based on the 1999 ISO definition (ISO/IEC 9899:1999 as corrected by ISO/IEC 9899:1999/COR 1:2001, ISO/IEC 9899:1999/COR 2:2004, and ISO/IEC 9899:1999/COR 3:2007). In the MISRA C:2012 guidelines, the vision of this third edition of MISRA C is stated to be to:

- adopt the 1999 ISO definition of C, while retaining support for the 1990 definition;
- correct any known issues with the previous edition;
- add new guidelines for which there is a strong rationale;
- improve the specification and rationale for existing guidelines;
- remove any guidelines for which the rationale is insufficient;
- increase the number of guidelines that can be processed by static analysis tools;
- provide guidance on the applicability of the guidelines to automatically generated code.

The *MISRA C:2012 Amendment 1 Additional security guidelines for MISRA C:2012* is an incremental update that extends the MISRA-C:2012 rule set. This Amendment adds additional guidelines aiming to improve the coverage of security concerns highlighted by the ISO C Secure Guidelines.

AUTOSAR C++14 specifies coding guidelines for the usage of the C++14 language in safety-related and critical environments, as an update of MISRA C++:2008, based on other leading coding standards and the research/analysis done by AUTOSAR. The main application sector for these guidelines is automotive, but it is also applicable in other embedded application sectors. The AUTOSAR C++14 coding rules address high-end embedded micro-controllers using POSIX or similar operating systems. For the ISO 26262 clauses allocated to software architecture, unit design and implementation, the proposed coding guidelines provide an interpretation of how these clauses apply specifically to C++.

In April 2016, MISRA published *MISRA Compliance:2016*, which provides enhanced guidance on achieving compliance to MISRA C and MISRA C++.

2.2.5 Functional Mockup Interface (FMI)

The Functional Mockup Interface (FMI) standard originated from the MODELISAR ITEA project, with the goal of ensuring a generic coupling of heterogeneous dynamic systems, using a common and independent interface. The FMI interface have been taken over by to MODELICA association (https://www.modelica.org/) as the FMI project. This standard, now widely used by tools providers, constantly evolves, steered by a college of industrial and academic members, including Siemens.

In AQUAS, the FMI standard is used to interface controller's binary codes with physical models and scenarios to achieve combined Safety-Security-Performance analyses of the overall Cyber Physical Systems.

2.3 Other relevant international standards

Most of AQUAS is concerned with domain specific and transversal standards that are mainly relevant for tools. However, among the standard evolution goals AQUAS intends to promote awareness and bring results into international standards in the functional safety and security area with respect to

safety, security and performance co-engineering, and influence actively standardization groups focused on frameworks for the coordination of safety, security and reliability of automation

2.3.1 Automotive sector standards activity

AQUAS does not have a use case in the automotive sector. However, there is vigorous ongoing standardisation activity, in great measure due to commercial pressure resulting from the race towards vehicle autonomy, advanced intelligent transport system applications, and resolution of automotive cybersecurity issues.

The automotive standardisation landscape covers many topics, from single-vehicle functional safety (ISO 26262) to ISO/SAE 21434 (Cybersecurity engineering) and Extended Vehicle (ExVe) (Extended Vehicle is addressing V2X, from to remote diagnosis to Time critical ExVe Road and Extended Vehicle Safety (RExVeS)). To establish a mid- to long-term ISO TC22 Roadmap towards Automated Driving, an ISO AG1 (ADAG – Automated Driving Ad-Hoc Group) was founded 2018. Most of the standards (except ISO 26262 and ISO PAS 21448, SotiF (Safety of the intended Functionality)) are under development and have severe safety and cybersecurity impact. **Figure 5** provides an overview over the automotive standardization landscape.

The ISO TC22 AG1 (ADAG, Automated Driving Ad-hoc Group) integrates work of several subcommittees of ISO (SC31, SC32, SC33 (ADAS), SC39 (Ergonomics UI), TC 204 (ITS), TC 241 (Road safety)), but keeps contact also to SAE, ETSI, CEN/CLC and IEEE.

Figure 5 Automotive standardization landscape ISO TC 22, ETSI

Of particular relevance to AQUAS is the remarkable fact that the current automotive standardisation activity involves three standards in evolution *at this time* (Autumn 2019) that address exactly the **three dimensions** treated by AQUAS and require interactions between these properties:

- **Safety**. The first version of ISO 26262 was published in 2011. While the standard was a huge success and adapted by the automotive industry, technological developments like the increased usage of assistant functions, increased connectivity and the rising importance of software required a revision and update of the standard. A second edition has been published in December 2018. The new edition contains a section on the interaction between safety and security and a requirement to define communication channels between safety and security. [13]
- **Performance**. Safety of The Intended Functionality – SOTIF: For automated or autonomous vehicles safety is not only endangered by failures in the classical sense, e.g. a hardware element is failing, or a software has a design error, but also by misinterpretations of sensor signals or lacking combination of sensor data and processing. SOTIF is a newly developed standard (ISO PAS 21448 – Publicly Available Specification) which addresses such issues. Of special interest to AQUAS is the fact that **inadequate performance** is explicitly considered in the standard as having potential impact on the other dimensions, in particular safety. This is nearly unique in the current standardisation landscape (although it is likely to become increasingly important, due to the automation of applications in diverse sectors from avionics to robotics and other related transport sectors). The PAS has been published in 2019. [14]
- **Security**. Due to increasing connectivity, V2X communication and the shift of functionality towards software and more complexity that increases the need for Over the Air Updates (OTA), cybersecurity is increasingly important for dependable automotive systems. Recently demonstrated hacker attacks on automotive control systems via maintenance or entertainment channels have highlighted the necessity as well. Therefore SAE, who created already SAE J3061 as guideline for Automotive cybersecurity engineering, and ISO have joined forces towards an Automotive Cybersecurity Standard (ISO/SAE JWG1, ISO TC22 SC32 WG 11, for ISO/SAE 21434). The standard has been scheduled for publication in 2020. Similar to ISO 26262:2018 ISO/SAE 21434 should contain a section for the interaction between safety and security.
- Another evolving cybersecurity-related standard concerns “Software update (conventional and OTA, Over the Air), ISO TC 22 SC32 WG12 ISO/NP 24089. It complements the other safety and cybersecurity standards.
- **Extended vehicle standards.** ISO TC22 SC31 has moved the extended vehicle standards, who handle the interaction between the vehicle and its environment (including other vehicles, road signs, sensors on other objects in the traffic flow, web interfaces, etc.) into a new WG 10 (time-critical extended vehicle applications). Sensor interfaces for automated driving functions remain in WG9. To harmonize somehow the standardization activities of various ISO committees towards automated driving and provide recommendations, TC22 has created an AG1 “Automated Driving Ad-hoc Group” (ADAG). In this context, security and safety interaction may become an issue as well.

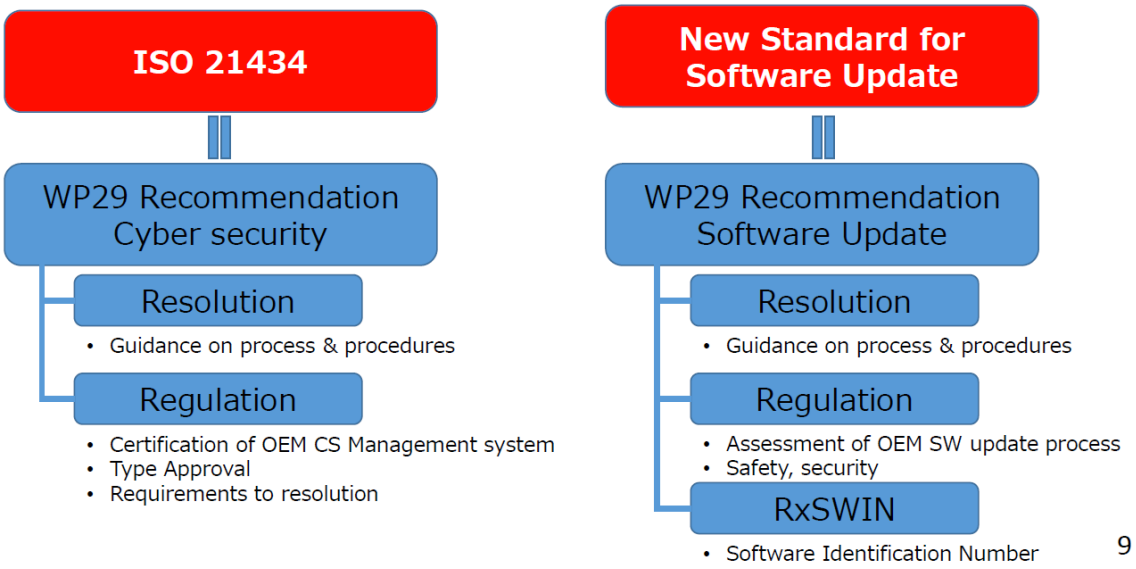

Figure 6: Automotive Cybersecurity engineering (ISO/SAE 21434) and Software Update Standard (ISO/NP 24089)

In addition, the regulations for road traffic are set by UN-ECE (United Nations Economic Commission for Europe), WP.29 (World Forum for Harmonization of Vehicle Regulations). UN-ECE WP.29 has recently drafted two new regulations, one for 'Automotive Cybersecurity Engineering', and one for 'Software Updates – OTA'. In both drafts, the upcoming standards shown in Figure 6 are already mentioned as example, so developers of these standards are certainly aware that they have responsibility for the future of road traffic and homologation of road vehicles.

WG 11 (ISO/SAE JWG1) has decided early 2019, after there have been more than 500 pages of comments on the CD1, and because the draft was already referenced by UNECE WP.29, to prepare a "Summer Baseline" of ISO/SAE 21434, which will be after the commenting phase be further developed towards a FDIS skipping the DIS stage. This "Summer Baseline" will be used as document for the testing of the new cybersecurity assessment for the type approval. This requires vehicle manufacture requesting a type approval to demonstrate that all participants in the supply chain are parts of a cybersecurity management system (CSMS) and to demonstrate cybersecurity of the vehicle.

ISO/SAE 21434 is intended as guidance for a CSMS and the cybersecurity case which is developed can be used to argue the cybersecurity of the vehicle.

2.3.2 Smart Manufacturing

It should be mentioned that the 'Smart Manufacturing' Standards are mainly covered by IEC groups of IEC TC65 (WG 23 and JWG 21, the latter together with IEC TC 184, Automation and integration). IEC TC65 WG23 is "Smart Manufacturing Framework and System Architecture for industrial measurement, control and automation" (this updated title and scope is currently under way). Here safety, security and reliability play an important role, often tackled in separate Task Forces on Cybersecurity, developing separate reports. Here is another "Window of Opportunity" open because of many ongoing standardization activities.

A key working document is IEC TC65 WG 23 "Smart Manufacturing requirements for Cybersecurity". It considers cybersecurity challenges for smart manufacturing, Systems engineering aspects, the application of IEC 62443 to Smart Manufacturing and the relation to the ISO/IEC 27000 series, smart manufacturing security threats explained by various use cases, the smart manufacturing life cycle view on cybersecurity, and a summary of challenges like timely response to events, resource availability, identification, authentication control, data integrity and confidentiality, use control, and restricted

data flow. Some other work was done concerning “Multi-domain confidence framework challenges - Application to Smart manufacturing safety and security properties” (Bertrand Ricque).

A basic publication is IEC PAS 63088:2017 Ed. 1.0 “Smart manufacturing - Reference architecture model industry 4.0 (RAMI4.0)”.

Related work is done in IEC TC65 WG16 “Digital factory”, with several standards evolving.

The notion of using “Digital Twins” gains more and more popularity. Digital twins are real-time digital images of physical objects, or processes, that are optimizing performance particularly in smart factories (see Figure 7). Many IEC committees are covering partial aspects of “Digital Twin”, particularly TC 44 (Safety of machinery – electrotechnical aspects, IEC TC65 Industrial process-measurement, control and automation (with WG 23, JWG21 with ISO TC 184, and WG 16 “Digital factory”).

The Joint Technical Committee JTC1 between ISO and IEC in the field of Information Technology has just recently started in the Joint Advisory Group for Emerging Technologies and Innovation (JETI) a new JTC1 Advisory Group (AG) “Digital Twin”, which should coordinate work in relevant subcommittees, e.g.

- ISO/IEC JTC 1/SC 27: IT security techniques
- ISO/IEC JTC1/SC 41: Internet of things and related technologies
- ISO/IEC JTC1/SC 42: Artificial intelligence

In ISO TC 184 SC 4, Industrial data, a new WG 15 is caring about ISO NP 23247, Digital Twin Manufacturing Framework ISO 23247 (TC184 SC4 WG 15). In risk management, safety and security as well as performance are expected to play an important role and AQUAS partners should monitor this development.

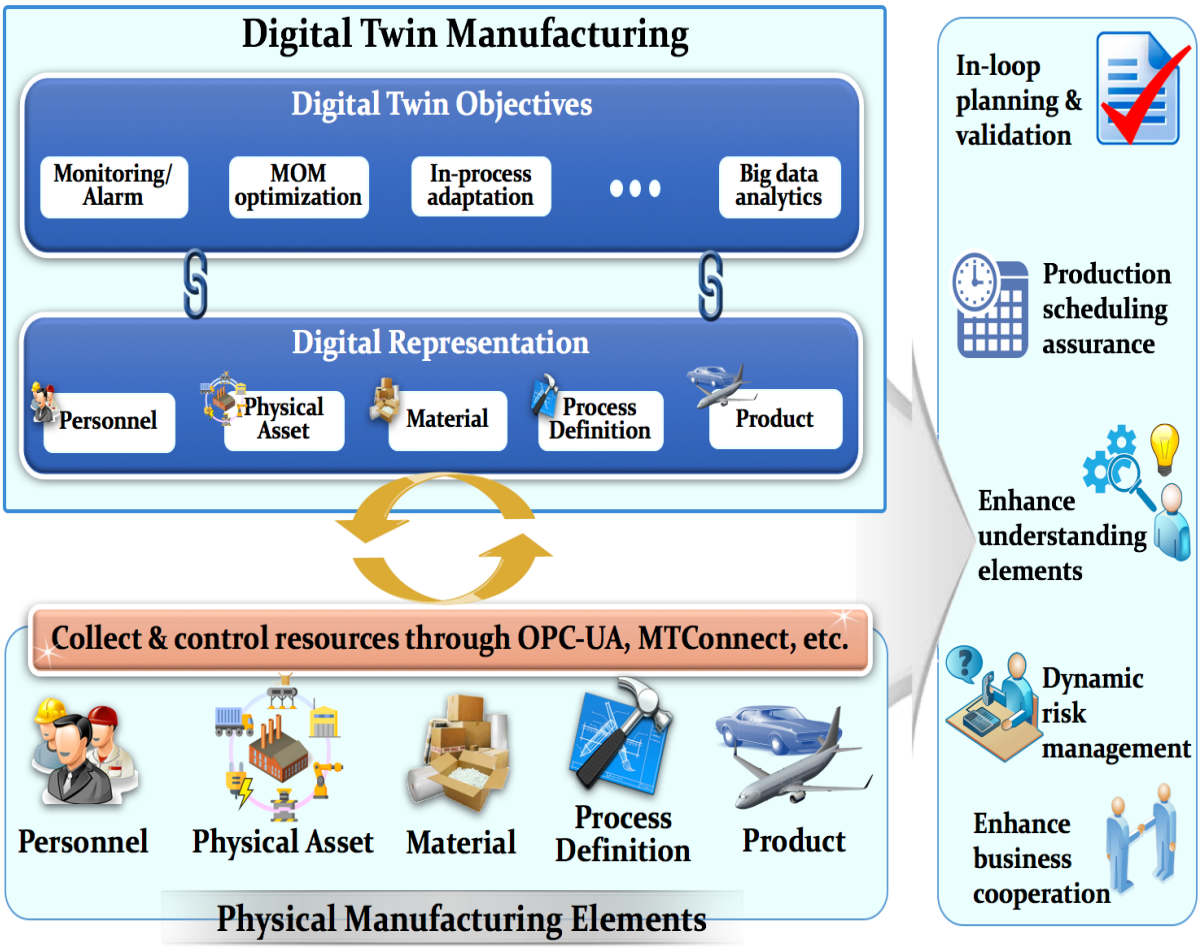

Figure 7: Digital Twin Manufacturing Framework (Source: ISO TC184 SC4 WG15)

2.3.3 Robotics

In ISO TC299, Robotics, there are several working groups on Industrial robots safety (WG 3), Service robots safety (WG 2), Service robots performance (WG 4), Service robots modularity (WG6), and a Joint ISO/TC 299 - IEC/SC 62A - IEC/SC 62D WG on Medical robot safety. Cybersecurity was no issue in the first versions of industrial robots' safety, but for the current revision, AIT (Austria) proposed among others to consider cybersecurity as threat to safety also in robotic standards. In the current maintenance phase, cybersecurity is already considered and also commented on, but mainly by referencing to IEC 62443, the IACS security standards. But to consider this security standard is now a mandatory requirement in ISO 10218-1, citation:

> "*5.1.10 Cybersecurity*
>
> *The robot shall be developed in compliance with the secure development lifecycle requirements as described in IEC 62443-4-1. The robot should be developed with consideration of a target Security Level as defined in IEC 62443 3 3. The robot shall comply with the applicable Security Level requirements in IEC 62443-3-3.*"

2.3.4 Adaptive Open Systems

Adaptive systems standards for assurance of dependability have been developed in IEC TC56, Dependability:

- IEC 62853/Ed1/CD © IEC:2015, Open systems dependability, and
- IEC 62741/Ed1: "Reliability of systems, equipment and components. Guide to the demonstration of dependability requirements. The dependability case". This standard defines a generic dependability case and assurance process.

These standards are not so well known in the safety and security community, but the methodology (process) can be applied to 'trust cases' as well.

2.3.5 Internet of Things and related technologies

As a specific example, consider framework-oriented standardisation for the Internet of Things. In the area of IoT [23], ETSI and AIOTI are working jointly on standardization, with particular focus on communication. Here security plays a dominant role, and therefore the challenge in this context is to bring "safety" into some fundamental statements. ETSI has just now established a Specialist Task Force STFCI (TC SmartM2M) on Security/Privacy and Interoperability of standardised IoT Platforms to fill a gap in the standardization landscape. In the field of embedded AI, ETSI has just now started an Industry Specification Group (ISG) on Securing Artificial Intelligence (SAI). This demonstrates the need to look into more into multi-concern challenges as well, to avoid too much focus on security only.

In ISO/IEC JTC1 SC41, *Internet of things and related technologies*, a joint committee of ISO and IEC in areas of common interest, Framework and Architectural standards (WG3), and Interoperability standards (WG4) are arising, besides WG5, Applications. Besides particular standards for sensor networks and wearables, the following evolving ones are of general interest here:

ISO/IEC CD 21823-1 [Under development]
Internet of things (IoT) -- Interoperability for internet of things systems -- Part 1: Framework

ISO/IEC NP 30147 [Under development]
Information technology -- Internet of things -- Methodology for trustworthiness of IoT system/service

ISO/IEC NP 30149 [Under development]
Internet of things (IoT) -- Trustworthiness framework

Trustworthiness is a key issue; the term covers a broad scope of dependability-related properties:

(citation from IoT Trustworthiness presentation, JTC1 AG7):

"Trustworthiness, corresponds to the ability to meet stakeholders' expectations in a verifiable way. Characteristics of trustworthiness include, for instance, reliability, availability, resilience, security, privacy, safety, accountability, transparency, integrity, authenticity, quality, usability.
Trustworthiness is an attribute that can be applied to services, products, technology, data and information as well as, in the context of governance, to organizations."

This extends even the notion beyond multi-concern assurance as addressed in AQUAS. It is relevant in IoT and AI standards, including ETSI and CEN/CENELEC Focus Groups.

ISO/IEC JTC1 SC41 has a liaison with ISO/IEC JTC1 SC27, *IT- Security techniques*, which hopefully becomes effective in our sense. SC41 has a Study Group on Blockchain (Security!) (AHG 18) and a Study Group on Societal and human factors in IoT based services (AHG 17), besides other groups and liaisons.

One of the most important one in international context is AG20 Sectorial Liaison Group (SLG1) on Industrial IoT (IIoT), since the IIC consortium has already established industrial IoT standards which should be somehow compliant with SC41 standardization work. A group AG22 Liaison Coordination Group (LCG) on IoT Trustworthiness was established to harmonize all issues of the different evolving JTC1 SC41 standards with respect to trustworthiness of IoT systems. The AHG19 Study Group on Swarm

intelligence for IoT (mass deployed devices and mobile objects) was unfortunately disbanded in 2019, although some progress has been achieved in several application areas.

2.3.6 Artificial Intelligence Technologies and Trustworthiness

With the evolving use of AI technologies in critical applications (safety, security, performance, whatsoever), the work on trustworthiness of AI applications becomes a severe challenging issue (for a definition of "Trustworthiness" see previous chapter). ISO/IEC JTC1 SC42 tries to standardize an AI and Machine Learning Framework, but without safety and security aspects being considered in the core of this framework at the moment. Here, the role and contribution of all partners is also to monitor the evolution of standards, particularly if trustworthiness, security and privacy issues should be addressed. Nevertheless, there are discussions under which conditions AI-driven components could be used in safety-critical applications or under security threats. JTC1 SC42 has the following working groups:

- ISO/IEC JTC 1/SC 42/AHG 1 Dissemination and outreach
- ISO/IEC JTC 1/SC 42/JWG 1 Joint Working Group ISO/IEC JTC1/SC 42 - ISO/IEC JTC1/SC 40: Governance implications of AI (Ethical considerations)
- ISO/IEC JTC 1/SC 42/WG 1 Foundational standards
- ISO/IEC JTC 1/SC 42/WG 2 Big data
- ISO/IEC JTC 1/SC 42/WG 3 **Trustworthiness (major point of interest for AQUAS)**
- ISO/IEC JTC 1/SC 42/WG 4 Use cases and applications
- ISO/IEC JTC 1/SC 42/WG 5 Computational approaches and computational characteristics of AI systems

2.3.7 Framework-oriented standards

Standards are sets of requirements that have to be met in order to consider something "compliant". **Frameworks**, in contrast, are "merely" sets of best practices and reference models. They can be used in the generation of domain-specific standards; they can be used in the absence of agreed standards; they can be used in order to help harmonize different standardisation activities. Indeed, framework-oriented standards are becoming more and more important in the standardisation environment for all of these reasons, and in particular the last – their ability to contribute to harmonization of multiple standardization activities. This is why AQUAS set an objective to also target framework-oriented standardisation activities for potential influence.

Elsewhere in this document examples of framework-oriented standardisation activities in systems engineering (PMBOK, Arcadia) and the IEC 61508 standard (see e.g. Section 3.2.2) are presented.

2.4 Human Factors

Human factors have essential roles in safety, security and system performance. But similar to safety and security, they are generally covered by specific professional groups and specific standards. Thus, in line with AQUAS objectives it was important to analyse these specific standards for such essential relationships. This emerged in the medical use case.

Additionally, it was identified in IEC 61508-3 preparation for Edition 3 that human factors are insufficiently covered in context of functional safety. So a working group was started in IEC SC65A, WG 17, for IEC TS 62879, "Human factors – functional safety". AQUAS partner AIT initiated that human factors do not only impact safety in the conventional manner as described below, but that particularly "security" has a strong impact and is mainly endangered by human interference ("hackers"). This is definitely influenced by the AQUAS co-engineering concept, and now in an early stage of drafting.

An update proposal for the draft TS 62879 on security impact and human factors on functional safety is currently under development, partially including considerations from ISO 27002 on human factors. The parts relevant for IEC 61508 Ed. 3.0 with respect to complement the existing references to human

factors in the basic functional safety standard IEC 61508 have been extracted and worked out as a proposal to the Maintenance Groups of IEC MT 61508-1/2 and IEC MT 61508-3 for further consideration..

In the medical domain, we identified several gaps and defects in current human factors/usability standards. This section outlines our observations and some possible remedies which could take the form of a commentary in the standards that designers can use to improve device safety and security, while being a possible proposal for updating the standards. A more detailed report for publication is currently in preparation. This section summarizes its major contributions.

Although we analysed many of the standards outlined in Section 2.2.2, our observations relate specifically to the following standards:

- EN 62366-1:2015: "Medical devices - Application of usability engineering to medical devices"
- IEC/TR 62366-2:2016: “Guidance on the Application of Usability Engineering to Medical Devices”
- EN 60601-1-8: 2007: "Collateral standard: General requirements, tests and guidance for alarm systems in medical electrical equipment and medical electrical systems" **including the recently proposed amendment EN 60601-1-8/AMD2 ED2.**
- **EN 60601-1-6: 2010. Medical Electrical Equipment. General requirements for basic safety and essential performance – Collateral standard: Usability**

Conceptual gaps in the standards that can lead to tunnel vision

We have observed some conceptual gaps in the medical standards. These are areas we highlight where the current wording of the discussion in the standards can lead to tunnel vision by readers: while highlighting some problems and solutions, it may cause readers to ignore others. Two of the major conceptual gaps we aim to address in our proposals for changes to the medical usability standards are described below in Sections 2.4.1and 2.4.2: (1) effects of usability on medical device security, and (2) causes of use errors beyond shortcomings in user interface design.

2.4.1 Lack of Consideration of Effects of Human Factors on Security

Usability directly affects device safety, performance and security, and although the former two are somewhat discussed in the standards, the relationship between usability and security is often overlooked. This is of concern. In particular, trade-offs between safety, security, and performance are in fact decided by the effects on human behaviour of various design decisions. If the task of taking into account human nature is left to specialists in human factors as a stand-alone task, these specialists may be able to improve "usability" in a narrow sense (e.g. fonts, colours, size of buttons) but these most important trade-offs would be decided almost inadvertently through design decisions (about hardware, algorithms, configuration details, procedures of use) taken without the necessary joint consideration of all effects of system design.

The relationship between usability and safety is especially important; “the majority of medical device incident reports can primarily be attributed to use error” [54]. Emphasizing this role adds to the significance of usability, which some designers may consider a minor issue. For example, displaying dose limits on a user interface display not only “can reduce the burden on the users’ memory and increase their confidence when programming the pump”, but also help prevent a harmful dose [IEC 62366-2].

Another significant relationship mentioned in the medical standards is that between usability and performance. As an example of this relationship, IEC 62366-2 discusses how high task performance might increase the safety of a device as it prevents delay of urgent therapy, but that it might also introduce new hazards if critical confirmation steps are not incorporated. On the other hand, slow

task performance could "lead a well-meaning user to pass over steps in a procedure to increase speed of the procedure. This can result in a higher probability of use error linked to a potentially unacceptable risk".

Besides the strong relationships between usability, safety and performance, new advances in medical devices have introduced another important relationship, which is often overlooked: the relationship between usability and security. Recent discussions in the AQUAS project between security and human factors experts highlighted such trade-offs. For example, user satisfaction increases when users are offered a wireless medical device compared to one with several connective wires that make it harder to use and move; however, a wireless design also introduces a range of new security vulnerabilities. As another example, security of use of a device may be enhanced with use of user authentication that prevents malicious/unintentional use; however, this may also prove to be a nuisance to some users, especially if repetitively required. Most importantly, in case of an emergency, authentication may inhibit a clinician's ability to respond in a timely manner, thus posing a safety hazard. This exemplifies our above-mentioned concern that usability-related decisions are really about crucial trade-offs between system properties, and thus require full combined analysis of all their effects.

This latter example depicts how all four factors: safety, security, performance, and usability may interact and influence manufacturers' design decisions (with regards to the level of authentication they may require as part of their device design). *We suggest that, similar to other relationships, discussion and examples in the standards of the human factor aspects of security may help designers consider and prepare for possible trade-offs that may arise.*

Also important to note is that such considerations of the effect of usability and relevance of human factors on other important qualities are best explored in the early stages of development and revisited later in the project, rather than only retrospectively considered when the device is in its later stages of development or, worse, in use. Early inclusion of such analyses is likely to guide important design decisions at the start when changes are less costly.

2.4.2 Incomplete Consideration of Causes of Use Errors: Beyond Shortcomings in UI Design

While the standards we examined document advances in the understanding of the effects of usability on safety and performance, and thus the responsibility of designers to address usability concerns, they often take too narrow a view of the causes of use errors. The standards repeatedly focus on the importance of user interfaces in a somewhat narrow sense: readability of displays, likelihood of confusion between buttons, etc. This is the area of causes of slips, typically (accidental errors in the execution of well-planned actions), but higher-level mistakes (misunderstandings of a situation and of what must be done) and intentional violations of procedures are also important hazards, affected by design and thus to be addressed in design and validation.

An important, recurring term in the usability standards is *use error*, defined as "user action or lack of user action while using the medical device that leads to a different result than that intended by the manufacturer or expected by the user" [EN 62366-1]. As explained in the standards, the term *use error* "was chosen over the more commonly used terms" *user error* or *human error* because not all such errors are "the result of oversight or carelessness by the user" [EN 62366-1]. In fact, the standards suggest that it is inappropriate to start by blaming the users: "although human beings are imperfect, it is inappropriate to blame the user when problems occur during summative evaluation. The key in any analysis of use errors, close calls or use difficulties is to intensely search for a design-based root cause before attributing the use error to the user." [IEC 62366-2]

We agree that renaming user errors to use errors is certainly a positive change as it removes judgment from the user. However, current explanations now seem to shift much of this blame onto user interface (UI) designers alone, so that other aspects of design are comparatively under-emphasised.

This idea is repeated frequently throughout the standards: “much more commonly, use errors are the direct result of poor user interface design” [EN 62366-1], “user interface design shortcomings can lead to use errors” [IEC 62366-2], and “the application of usability engineering is a principle means to reduce medical device unacceptable risk and improve patient care by reducing the potential for harmful use error through enlightened user interface design” [IEC 62366-2]. Even the definition of usability as being “characteristic of the user interface” points in this direction [IEC 62366-1].

It is certainly true that non-intuitive displays, hard-to-learn controls, confusing menus, or ambiguous alarm signal messages - all examples of user interface design flaws - may lead to use errors. However, abundant experience, research literature and medical device incident reports reveal that these are not the only causes. Thorough identification of use-related hazards must consider: (1) the users, (2) the use environment, (3) the device design (including the user interface) and (4) the complex interactions between them. The diagram below lists examples of factors in each of these categories, and depicts how the user interface is just a single player in a web of potential causes. *Not all of these causes are effectively discussed in the standards, and too much focus is on user interface design.* Quite often the environmental causes of use errors are in procedures, user workloads, responsibilities, etc., which while not under direct control of the designers are affected by documentation and labelling recommendations that may need to be informed by the supplier.

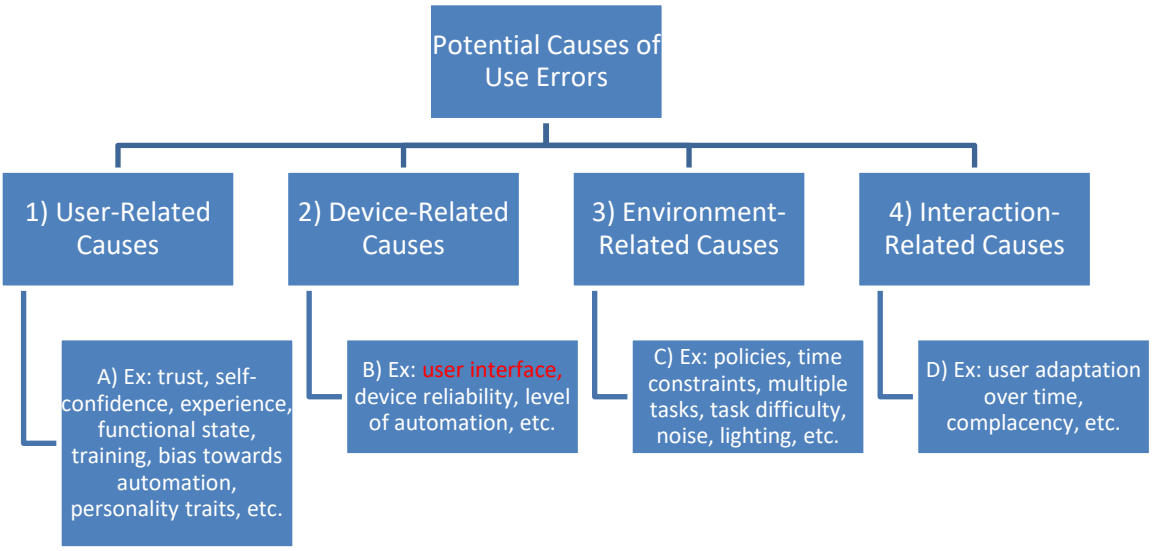

Figure 8: Potential causes of use errors beyond shortcomings in user interface design

As an example of how various factors in Figure 8 may interact to trigger a use error, consider a scenario where a physiological patient parameter has reached a dangerously low level and thus warrants immediate user action. To start with, user action is likely influenced by whether the device algorithm is designed to detect this danger with high enough probability (i.e., tool reliability – Box 2B in Figure 8). In case the device does detect this danger, the alarm signal emitted then needs to be visually and/or aurally communicated effectively to the user (i.e., user interface – Box 2B in Figure 8). However, in order to prevent a hazardous situation, it does not only matter whether the alarm is clear and audible, but also whether in practice it will lead to correct user action (with high enough probability). This may depend on environmental factors such as whether the user is busy dealing with another, simultaneous task (i.e., multiple tasks – Box 3C in Figure 8). It also depends on user factors such as the user’s *mental model*: their “conceptual model of how the [device] works and is structured” [EN 60601-1-10]. In turn, mental models are based on users' knowledge and thus depend not only on their training (Box 1A on

Figure 8) but also on their experience of interaction and learning curve when using the device (i.e., user adaptation – Box 4D in Figure 8).

Just as the standards have moved away from placing the blame entirely on the user, we argue it is inappropriate to shift this blame onto the user interface designers. While in the standards there is a need to focus on the role of sound user interface design to ensure that designers take certain precautions, designers who focus solely on the role of user interface are likely to overlook other causes and thus fail to adopt mitigations in their designs to address these causes. We use Table 1 to help illustrate the danger of such tunnel-vision. The first two columns describe specific use errors and user interface design shortcomings that may cause them and are taken directly from the standards [IEC 62366-2], except for text in italics which indicates authors' comments. We add the third column to illustrate other plausible non-user interface causes of the same use errors, and the fourth column to introduce possible remedies against these various non-user interface-related causes.

Use Error	User Interface Design Shortcomings	Other Possible Causes Not Related to User Interface Design	Potential Mitigations Addressing the Other Possible Causes
Users fail to detect a dangerous increase in heart rate because alarm limit is set too high and users do not look at medical device display because they are over-reliant on the alarm system	User-adjusted high and low alarm limits on a heart-rate monitor are not continuously displayed *(implicit solution: continuously display alarm limits)*	User chose inappropriate alarm limits either due to inexperience (user-related cause; Box 1 in Figure 8) or in an effort to reduce the device alarm rate which they find distracting (device-and interaction-related causes; Boxes 2 and 4 in Figure 8)	Consider how the alarm threshold (sensitivity/ specificity combination) is set – not just choosing a more/less sensitive rate, but also considering default settings, degrees of freedom by users, and customization according to certain attributes such as user ability.
User ignored a warning label telling the user to disconnect the patient tube before turning the medical device off	The medical device did not require the user to confirm patient disconnection before powering-off *(implicit solution: add a verification step to confirm patient disconnection before powering off is allowed)*	User, at the end of a long medical procedure, is fatigued and overlooks the importance of this step (user-related cause; Box 1 in Figure 8). Or other devices, which the user is accustomed to, dictate that equipment must be turned off before disconnecting from the patient (environment-related cause; Box 3 in Figure 8).	Add a verification step to confirm patient disconnection before powering off is allowed. Redesign the device so that the order of these operations does not matter.
User disregarded a warning symbol and allowed a portable medical device to run out of battery power	The warning symbol was not sufficiently attention-getting *(implicit solution: make the warning symbol more visible/audible to attract the user's attention)*	Lack of reaction to an alarm due to factors such as the "cry wolf" effect[8]. Paradoxically, designers can make devices more sensitive only to find that user decisions become less sensitive[9]. In other words, it may not be that a user did not see/hear the warning, but that their experience with the device has led them to ignore it (interaction-related cause; Box 4 in Figure 8).	Ensure that the time between when an alarm is emitted and the actual danger occurs is optimally chosen in a way that does not cause the user to ignore the alarm and delay action, but still gives them sufficient time to react. During user training, raise awareness against behaviours such as "cry wolf". Consider potential unwanted interactions between different alarms, their effects on user reactions, and their grouping or prioritization to reduce such effects.

Table 1: Various Causes of Use Errors -*Source: IEC 62366-2*

As can be seen from the third column, our concerns are not denying the role of effective user interface design, but emphasizing that some use errors can be the result of other user, environmental, or interaction-based issues. Table 1 highlights that although some of these other causes can perhaps be remedied using the same design mitigations that address interface-design shortcomings (such as in

Row 2), some of them require different strategies (such as in Rows 1 and 3). In fact, in Row 3, making the warning symbol more attention-getting not only does not address the "cry wolf" effect discussed in the third column, but may even exacerbate it. This highlights the importance of taking a holistic approach to analysis of the causes. Finally, Table 1 also illustrates how many mitigation strategies may extend beyond changes to the user-interface and may instead address the user or environment.

This important category – errors in response to alarms – of use errors, is a good illustration of how the recommendations in the standards may be simplistic, rightly highlighting the risk from very basic design flaws but not alerting designers to subtler causes of the same errors that also require care in design. The complex interactions of multiple factors that can lead to use error of this kind are exemplified in the publication by Alberdi et al. [55], which documents multiple causal chains that may lead to these use errors. By heeding these more extensive explanations of use errors, the standards could avoid a narrow focus on user interface alone and protect against tunnel vision in design.

Figure 9 shows an analysis of possible causes, *not linked to user interfaces*, of non-response to alarms, (from [28]). The graph is meant to assist designers in identifying the causal chains leading to undesired effects so that they can interrupt the chain with appropriate mitigations.

[8] "Cry wolf" effect: users failing to intervene when they should, because a high rate of device false alarms trained them to ignore alarms

[9] The easiest way to picture this effect is in the context of smoke alarms. The more sensitive they are to smoke, the more false alarms they emit, and the more likely people are to ignore or even disable them.

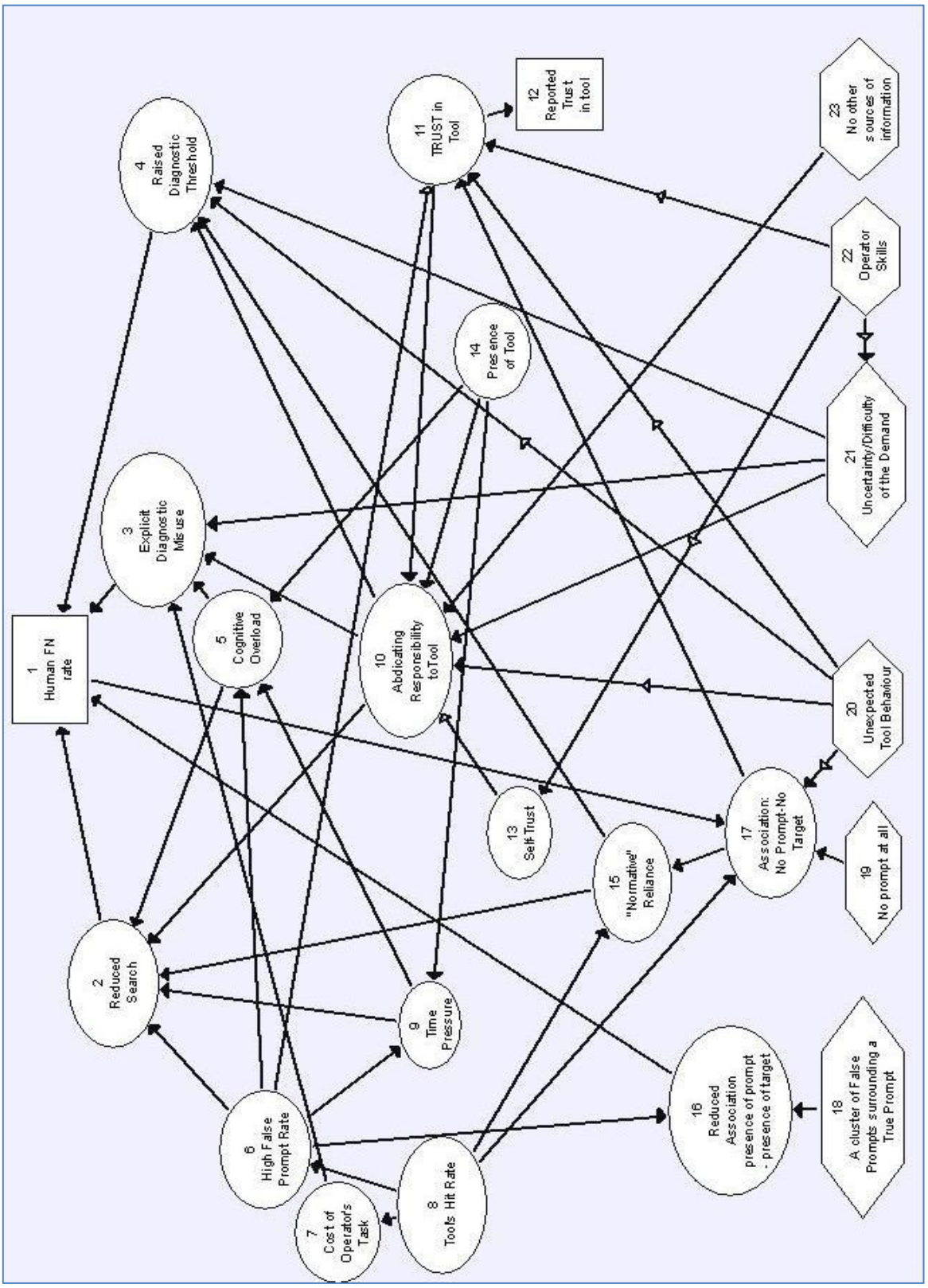

Figure 9: Analysis of possible causes of non-responses to alarms

We note that the recently proposed amendment to EN 60601-1-8 shows a step in this direction by introducing new terms such as *alarm fatigue, nuisance alarm signal* and *alarm flood.* However, we argue that even when some of these causes are mentioned in the standards, the focus remains on user interface design, and we have shown examples of the danger of such a restricted view both when

considering causes and/or remedies. We welcome and look forward to the adoptions of the proposed EN 60601-1-8 amendment. Naturally, similar changes will likely need to follow in the collateral usability standard itself: EN 60601-1-6, and also EN 62366. Most importantly, however, the definitions in the amendment are a first step. Examples would help readers understand the implications of these effects, their magnitude and frequency, and how they may be manifested in real domains; especially examples that reveal less obvious issues such as: (1) how a desirable design goal can lead to a use error when considered in isolation, (2) how factors can interact and trigger one another to lead to a use error, and (3) how users can be affected by a device without actually conforming to its advice. The overall focus of the standards also needs to address these newer causes, such as the argument presented in Table 1. Finally, presentation of the issue is most useful if, hand-in-hand, mitigations are also presented to help reduce and possibly eliminate such causes of use errors, with special consideration of the effect of time on user's mental models. This latter point is another key gap we focus on in our detailed proposals.

We emphasize that this holistic approach to potential causes of use errors is becoming increasingly important. Medical devices are now used with increased frequency in busy environments, with new distractions. Also, patient care is evolving, sometimes moving to new environments such as private homes, or sometimes allowing use by less skilled or even unskilled users and patients; thus, wider considerations that incorporate such factors are becoming more important than ever.

IMPROVED CLARITY AND CONSISTENCY OF DEFINITIONS

The examples of conceptual gaps discussed above represent high-level observations of the medical standards; equally important are lower-level observations regarding the definitions of specialized terms used in the standards. Precise and consistent definitions ensure a common understanding of the standards by all the different people involved: managers, engineers, researchers, clinicians, business professionals, etc. Holistic and thorough definitions can also help highlight causes or consequences of hazards that may otherwise be overlooked. Our analysis of the medical standards reveals that crucial concepts such as *alarm system, alarm condition* and *alarm signal* are defined in ambiguous/inconsistent ways that may lead users of the standards to overlook certain hazards [EN 60601-1-8:2006]. Also related to the idea of definitions is how they affect the scope of the standards which is currently limited to assessing and mitigating risks caused by normal use, and helping identify but not assessing or mitigating risks associated with abnormal use [EN 60601-1-6], where abnormal use is defined as, "conscious, intentional act or intentional omission of an act that is counter to or violates normal use and is also beyond any further reasonable means of user interface-related risk control by the manufacturer" [EN 62366-1:2015]. Using a counterexample, we discuss how this definition of abnormal use may encourage the exclusion of certain design decisions that may cause a user to deliberately go against appropriate use; and thus are not necessarily "beyond any additional means of risk control by the manufacturers" and should not only be assessed but also mitigated.

LATEST METHODOLOGIES AND TECHNIQUES FOR EFFECTIVE USABILITY

An important part of the standards is the summary of the latest research on accepted techniques and methodologies that may be used to achieve effective usability. We note a thorough discussion of a variety of techniques. However, we also note an over-simplification of important concepts that should be considered**, especially regarding the estimation of the probability of occurrence of a use error,** which we address in our proposals, supported with more recent references.

2.4.3 Human Factors in Automotive Standards

Within the automotive standards, there is work in human factors in the context of the SOTIF standardisation work described previously.

Driver interaction with a highly automated vehicle is clearly very sensitive to human factors, and therefore has been brought into the scope of the SOTIF. The approach is based upon the *Human Factors Analysis and Classification System* (HFACS) [58].

An example of the approach is presented in Table 2.

Performance limitation scenario	**1) Stakeholders**	**2) Misuse causes**		**3) interactions between driver and system/vehicle**	**Misuse scenario 4) consider condition of environment**
		process	**Guide words**		
"While operating autonomously on a highway, the vehicle cannot estimate the location of the lane boundary due to a performance limitation. The vehicle starts to leave the lane and the driver is notified to take control."	Driver …	Recognition	1. Do not understand	Operation (Usage)	…
				Vehicle behaviour	…
				Warning/information	"Driver does not take over control of the vehicle and vehicle departs lane because driver does not know meaning of the warning"
			2. False recognition	Operation (Usage)	…
				Vehicle behaviour	…
				Warning/information	…
		Judgment	3. Judgment error/ misjudgement	…	…
		Action	4. Slip/Mistake	…	…
			5. Intentional "driver vacated seat"	…	…
			6. Unable "Driver not paying attention Driver asleep"	…	…
	…	…		…	…

Table 2: Examples of misuse scenarios in the SOTIF

Although AQUAS does not have an automotive use case, this demonstrates the general interest of this dimension across mission-critical domains.

2.4.4 Human factors in Space Standards

In the ECSS family of standards, ECSS-E-ST-10-11C – *Human factors engineering* (31 July 2008) forms part of the System engineering branch of the Engineering area. As such it is intended "… to assist in the consistent application of human factors engineering to space products by specifying normative provisions for methods, data and models to the problem of ensuring crew safety, well-being, best performance, and problem avoidance in space system and payload operations".

Note the link to both safety and performance in the descriptive text of the standard – more evidence of the relevance of human factors to the AQUAS objectives.

3 Co-Engineering Gap Analysis of Current Standards

3.1 Current co-engineering issues in standards development

In multiple domains, which already have safety as an established property, security is becoming a new issue. Due to increased interconnectivity and usage of Commercial Off The Shelf (COTS) components in safety-critical systems there is an growing threat to the cybersecurity of safety-critical systems. Usage of COTS components leads to common vulnerabilities in different systems and more people with the skills to identify and exploit vulnerabilities. In addition, cyberattacks are increasingly used as a new method for covered attacks by state-connected or terrorist groups. Cooperation between such groups and semi-commercial hackers who search for and sell zero-day exploits and malware kits leads to threat actors with increased expertise and resources. This, combined with the increased attack potential, leads to a rising threat landscape for safety-critical systems.

The rise of autonomous systems in multiple domains has led to the need to include **performance** as part of the mix, since it has been observed that inadequate performance (e.g. of advanced sensors) can have an effect on the other dimensions, particularly of safety. In addition, it has been recently observed that measures taken to enhance performance have had serious consequences on the other dimensions. For example, the Meltdown and Spectre bugs arising from attempts to enhance the performance of Intel processors have seriously compromised their security characteristics.

The introduction of performance into co-engineering is extremely recent, and few standards are treating it to date (a notable exception is the automotive SOTIF). Thus, we must rely for now on experience with cybersecurity and safety dimensions in the standards developing organizations in order to see how they are currently addressing the topic of co-engineering.

In most domains there has been a long discussion about how to address this new challenge. The discussion has focused mainly on how to address the issue of security in safety critical domains in the standards. Discussed approaches include:

1. Use established security standards for security engineering in safety-critical domains
2. Extend established safety standards with security engineering in safety-critical domains
3. Develop own security standards for security engineering in safety-critical domains

Most domains decided on approach 3 while also integrating links from safety to security in their safety standard, which is very important, since cooperation between both areas is crucial for success. The following is a brief summary of the reasons why it was necessary to develop specific security standards for safety-critical domains. Since the development of security standards is still ongoing, this list does not include a solution to the approaches, but presents challenges when trying to apply “standard” IT security to safety critical systems or trying to use “standard” safety approaches for cybersecurity. As noted earlier, the inclusion of the performance dimension adds another element of complexity that has yet to be addressed in the standards developing organisations, and could become therefore an issue of interest to contribute for AQUAS.

3.1.1 Risk assessment and management

At first glance, risk management in cybersecurity and safety is similar. Based on an initial risk assessment measures for risk reduction or mitigation are implemented. During operation incidents are monitored and, if evidence shows that the risk management is insufficient, additional efforts are required. But while safety assumes a *random* distribution of failures over time and components, security needs to consider an *intelligent* attacker. Attacks are timed to maximize the impact and if an incident is detected the system is often compromised in multiple additional ways. In addition, safety relies for risk assessment on existing information about past systems to enumerate the risk and

determine an acceptable level. The combination of our limited experience of the new forms of advanced persistent threats and the growing interconnection of critical components reduces the usefulness of past experiences. Hidden interdependencies like reliance on common infrastructure (time or position server) or components (same variant of SW or encryption library) leads to single attacks which brings down many different systems. In previous work, AQUAS researchers have observed that it is inherently infeasible to associate some kind of probability with security risk, whereby it is not infeasible for safety. [15][16]

Current risk-management techniques from safety are blind to intelligent attackers and have no usable existing data. Security analysis misses the cyber-physical dimension, e.g. the impact on the real world and consideration of system environment. **Performance risk analysis** to date has been nearly entirely associated with nominal functional operation (often linked to commercial issues such as minimal acceptable performance by customers), with little or no relationship established to safety and security dimensions.

Therefore, the establishment of a unified risk assessment / management regime in co-engineering for all three dimensions remains a significant challenge.

3.1.2 Incident reporting and sharing

A common best practice in all dimensions is the recording of incidents – for example, for compiling "lessons learned". However, there are pressures of different kinds that inhibit incident sharing.

To date, the recording of **performance incidents** has been generally kept proprietary by the manufacturers when the incidents did not have a clear impact on either safety or security. This is generally in order to protect brand reputation – that is, to fix defects without adverse publicity.

In the safety dimension, while sharing of safety incidents increases the level of achievable safety for all, sharing of security risks can, in the worst case, increase the risk level for all. Especially for safety-critical legacy devices which are often not continuously connected closing of vulnerabilities is a time-consuming process. Publishing vulnerabilities leads to a "window of vulnerability" which can exist for quite some time. In addition, processes and responsibilities for sharing of vulnerabilities are currently in definition and not established in industry. Therefore, processes and responsibilities for cybersecurity incident sharing are not defined. Especially with shared components domain specific sharing can lead to risks to other domains. Existing security sharing policies cannot be copied to the safety domain. Existing performance related incident recording policies are generally unique to the manufacturing organization and often kept as proprietary as possible in order to protect IP and avoid negative commercial consequences. Now that performance is being related to safety and security impacts, this will have to change, while acknowledging the commercial pressures on manufacturers.

3.1.3 Safety / Security / Performance related development processes

Nearly every standard regulating mission-critical systems development specifies a development **process** (the left-hand side of the V-model) together with a set of recommended best practices according to the dimension being treated by the standard. For example, a standard focusing on performance might focus on algorithm quality or recommended performance benchmarks. A safety engineering standard might focus on a set of recommended "safety patterns" that are well trusted in the community. But they can come into conflict.

For example, the need to consider cybersecurity threats reduces the usability of established safety engineering patterns. Redundancy and diversity are well-established safety mechanisms. Software-based systems rely mainly on diversity, e.g. having two different versions of software or even systems for the same task. Considering cybersecurity this increases the potential attack surface and requires auditing and ensuring security of two supply chains, including checking all used COTS elements and vetting suppliers and involved developers. Established safety design-patterns lead to an increase in

cybersecurity risks and there are no easy solutions, either from an architecture or from a process side. New architectures and design concepts need to consider safety and cybersecurity. An example is the potential usage of cryptography for security (confidentiality) *and* safety (error detection).

On the other hand, diversity may be used to detect certain types of anomalies since it is more unlikely that both (or all, if more channels are available) channels are attacked by the same means. If multiple diversity is available, this would allow continued operation while the infected or disrupted channel is cleaned. Pure homogenous redundancy is prone to react in a malicious way to the attack at the same time.

Finally, redundancy is also a commonly used technique in order to ensure adequate **performance**, including availability (in terms of nominal functionality). But its interpretation in performance-related development is clearly different from that of safety and security. AQUAS researchers (especially CITY) have been particularly active in research in different types of redundancy and their effect on safety and security.

In summary, each of the three dimensions treats “best practices” in development in different ways, making it difficult to harmonize the standardisation of development according to each of the three dimensions. This remains a challenge for co-engineering standards.

3.1.4 Safety / Security / Performance related testing, analysis and V&V processes

Whereas the development process addresses the left-hand side of the classic “V model”, the testing, analysis, and verification / validation processes address the right-hand side. Standards such as the DO-178B (“Software Considerations in Airborne Systems and Equipment Certification”) emphasize the importance of software verification. Verification is defined as a technical assessment of the results of both the software development processes and the software verification process. Sec. 6.0 of the DO-178C states that “verification is not simply testing. Testing, in general, cannot show the absence of errors.” The standard consequently uses the term "verify" instead of "test" when the software verification process objectives being discussed are typically a combination of reviews, analyses and test. The purpose of the software verification process is to detect and report errors that may have been introduced during the software development processes. Removal of the errors is an activity of the software development processes. The general objectives of the software verification process are to verify that the requirements of the system level, the architecture level, the source code level and the executable object code level are satisfied, and that the means used to satisfy these objectives are technically correct and complete. At the code level, the objective is to detect and report errors that may have been introduced during the software coding process. The non-functional safety properties are explicitly listed as a part of the accuracy and consistency verification objective at the code level, including stack usage, worst-case execution timing and absence of runtime errors.

Performance testing is well aligned with classic testing and V&V, with a long history of structuring the process and automating the execution to reduce human effort and increase test coverage. While performance testing typically focuses on maximizing throughput or minimizing latency, safety-oriented performance analyses focus on ensuring compliance with predefined performance limits, in particular resource constraints such as stack size or real-time deadlines. A technique which can provide guarantees that such limits are met is sound static code analysis at the executable object level. Meeting resource constraints is a typical requirement of safety standards.

Safety testing increases complexity further, whereby different phases in the lifecycle rely on defined test approaches to ensure correct implementation of safety measures and sufficient risk reduction.

The situation changes with security. Security testing follows different strategies and the highest level is human testing (penetration testing) which is only partially structured and difficult to automate. Identifying and using overlaps between security, performance, and safety testing has the potential to reduce effort. One such overlap exists at the code level. A common safety requirement is that runtime

errors due to undefined or unspecified behaviour of the programming language must be prevented since they can cause erratic and erroneous behaviour and, hence, may provoke safety hazards. Examples are division by zero, buffer overflows, or data races. At the same time these programming defects also represent the most important security vulnerabilities at the code level which enable data leaks, code injection and denial-of-service attacks. Sound static analysis at the code level can detect all such runtime errors and constitutes a common activity in the safety and security life cycle.

AQUAS results in this area could become a valid input to standardisation efforts in the various domains.

3.2 Gaps in selected current standards

In the following sections, representative standards are selected for gap analysis, both as examples of issues in co-engineering, and possibly to be influenced for evolution.

3.2.1 ECSS standards

The ECSS standards family is chosen here as a representative of the problem of integrating separate, pre-existing standards for safety, security, and – in *some* cases – performance.

In Space projects, currently the standards for Safety, Security, and Performance are handled **separately**.

In the ECSS, dependability and safety are the only aspects that are handled to define the criticality of the software. For defining this, no Security or Performance aspects are considered. Safety and dependability standards are defined at system level separately in the ECSS-Q-ST-40 and ECSS-Q-ST-30 documents, while are handled together at software level in the ECSS-Q-ST-80 and ECSS-E-ST-40 documents (software quality and software engineering respectively).

It would be useful if the ECSS standards indicated a concrete set of security and performance requirements so that they can be used for the selection of the software criticality and tailoring.

For Space software projects, there are currently **two scenarios**: one scenario for which there are performance standards that can be used, and another scenario for which it would not be realistic to implement performance standards. Therefore, trying to introduce a specific performance standard across *all* Space projects would not be advisable. Regarding security, we believe standards are in the same situation as described for performance.

The ECSS software standards do point to dependability and safety standards. As we study co-engineering in the context of AQUAS, we believe it would be advantageous to have a link to a concrete set of security and performance standards in order to have all these aspects considered when defining the software criticality. Currently the standards and the development efforts are tailored according to the criticality of the software by means of an applicability matrix. Therefore, this tailoring would need to consider when safety and security aspects are applicable.

There has been a recent evolution of the current ECSS standards, where the definition of the criticality categories and how to perform dependability and safety engineering in a project have been revised and partially harmonized.

In addition, in July 2016, the ESA Board for Software Standardisation and Control released a first version of a "Secure Software Engineering Standard", ESSB-ST-E-008 and a companion "Secure Software Engineering Handbook", ESSB-ST-E-007. The purpose of these documents is to enhance the ECSS-Q-ST-80 and ECSS-E-ST-40 software development process in the area of secure software development. In particular the engineering security requirements within the ESSB-ST-E-008 standard are formulated as delta requirements on top of the E40 standard. They introduce the security viewpoint all along the standard, the same way as e.g. dependability is mentioned.

However, the ECSS standards still do not consider the co-engineering aspects *together* for the classification of projects according to criticality categories. Within AQUAS, we propose to aim our efforts in standards evolution towards this goal.

3.2.1.1 UC5 CE approach versus ECSS-E-ST-40C

In this section, some gap analysis of the ECSS Software Engineering Standard against AQUAS methodologies followed in Space Multicore Architectures (UC5) is made. Analysis is made only by comparison of those activities carried out in the frame of UC5 interaction points (IPs). In summary, the UC5 IPs have one or several co-analysis activities and are distributed across the PLC in the following manner as described in Table 1 (a preliminary IPs table was published in D3.2 [17], this update will be available in D3.3):

Table 3 UC5 Interaction Points

IP ID	CA ID	Description/purpose	PLC phase & "When"	Attributes studied
IP_Cph_1 (IP1)	CA1_1	Detection of requirements interferences. Tagging of requirements (according to S/S/P concerns).	Concept phase	Safety, Security, Performance
	CA1_2	Methodology selection regarding applying formal methods.		
IP_Dsph_1 (IP2)	CA2_1	Identify interferences due to safety-security barriers in the architecture.	Design phase	Safety, Security
	CA2_2	Refine functional architecture to include the safety and security requirements. Early schedulability analysis to evaluate performance		Safety, Security, Performance
IP_DsDvph_1 (IP3)	CA3	Safe Scheduling, Safe Code generation and performance analysis	Between design & implementation phases	Safety, Security, Performance
IP_DsDvph_2 (IP4)	CA4	Safe Scheduling, Safe Code generation and performance analysis	Between design & implementation phases	Safety, Performance

- **SW related System requirements process**. The first activity consists of identifying those system requirements allocated to SW. The tasks done here should not be restricted to safety, security and dependability analysis, taking into account other aspects as performance. In AQUAS UC5 interaction point IP1, interferences among requirements allocated to different concerns are identified. This is made in two steps: firstly, by tagging each non-functional requirement as concerning to safety/security/performance aspect, and then by adding inter-links between pairs of requirements that could interfere one another. Although this activity is

done in AQUAS UC5 for software specific requirements baseline, it could be extrapolated to System requirements.

In §5.2.3.1 and §5.2.3.2 of [ECSS-E-ST-40C] it is proposed to identify the requirements to be verified/validated from the requirements baseline; in AQUAS UC5, the approach is an analysis (by system engineer, SW engineer, and formal methods engineer) to decide the verification strategy (formal methods vs standard V&V) for requirements validation. Formal method strategy (provision of formal correctness proofs through "contracts" objects in the design model, or as part of the SW code) reduces effort in Implementation and V&V stages. These contracts can be defined considering several concerns. This analysis is performed in IP1 (static vs runtime verification method is also proposed).

In §5.2.4.7 of [ECSS-E-ST-40C], a list of requirements for "software to be reused" is to be given by customer. This approach could be coordinated with those requirements where formal methodology is applied, so reduction cost could be achieved in recurrent use of these requirements in different projects.

In §5.2.4.8 of [ECSS-E-ST-40C], the procedure is as follows: a) Perform analysis indicated in [ECSS-Q-ST-80], which are FMEA, FTA and common cause failure analysis, this is done by the customer over the System requirements (in the SSS-Software System Specification document); b) Determine the software criticality; c) Specify the software safety and dependability requirements, being the latter a supplier activity. In AQUAS UC5, interaction point IP2, a first co-analysis activity results in FTA including safety/security barriers, while the second puts these results into a refinement of the architecture design, to perform a new iteration of the schedulability analysis. These analyses are multi-concern activities (i.e. considering safety/security/performance view points). Input for these analyses is the requirements baseline (including safety / dependability / security /performance requirements). Available output artefact, in addition to the refined architecture model, is a new set of safety/security/performance software requirements (input for the SRS document).

In section §5.2.2.1 of [ECSS-E-ST-40C], section a, a list of requirements is provided by the customer. These artefacts (RB, SSS, SRR) already include requirements for real-time, security and performance, although security it is not mentioned, but included in ESSB-ST-E-008 amendments. While in AQUAS UC5, these requirements are also produced, also a co-analysis is done to detect potential interferences among these requirements. This information is cascaded to the next phases.

- **SW management process.**

Chapter §5.3 of [ECSS-E-ST-40C] deals with the SW life cycle management. In AQUAS UC5 the life cycle is enhanced by having IPs with several CAs. These IPs could be determined at the beginning of a project by performing them at some point in the PLC, and those artefacts generated in the IPs could possibly be included as part of the expected outputs in the different milestones of the project. For instance, as part of the joint review reports (section §5.3.3, the reports of the different CA could be included as parts of those reports (for SRR, PDR, CDR milestones). A proposal for UC5 artefacts mapping is shown in Table 4:

Table 4 UC5 Mapping to Joint Reviws

AQUAS UC5 Artefact	SRR	PDR	CDR
IP1_art	✓	✓	
IP2_art		✓	
IP3_art			✓

IP4_art	✓

§5.3.8.1 concerns management of software technical budgets (CPU, memory, deadlines, communications, throughput …) as specified in the requirements baseline. UC5 space multi-core architectures deals with these budgets in the design and implementation PLC phases (interaction points IP2, IP3, IP4). Being a (space) multi-core architecture, and considering existing hardware environmental constraints, these analyses are even more relevant. The CAs are mainly carried out considering safety & performance aspects, however the security profile in this case impacts the former safety-security analysis. The aim is looking for possible interferences, to be traced further in the PLC (see ESSB-ST-E-008, 5.3.8.1). Methods used in UC5 include schedulability (IP2) and time analysis (IP2, IP4), timing interference characterization (IP3), and safety/security co-analysis (IP2). Results of these analysis will derive in:

- update of the requirements baseline (imposing new/modified requirements to assure the technical budgets)
- modifications in the architecture models baseline
- code re-factorizations enhancing the safety, etc.
- preliminary performance results (e.g. sensitivity results regarding schedulability).
- security/safety analysis report (security issues analysis, FTA indicating issues regarding security requirements, performance/resource requirements analysis).

- **SW requirements and architecture engineering process.** §5.4.2.4 and §5.4.2.5 in [ECSS-E-ST-40C & ESSB-ST-E-008] mandates conducting a Software/System requirements review: “The customer shall update the security strength of function requirements to reflect the current understanding of the project specific security requirements”. For UC5, this is done in interaction point IP1 (dedicated to software requirements analysis) and IP2 (software architecture design). In AQUAS, the software requirements reviews are dedicated to several S/S/P concerns simultaneously, in search of possible interferences.

  In chapter §5.4.3 a software architecture is produced as a result of the transformation of requirements into a design. In IP2 several CA are performed to design an architecture that have safety and security barriers, and some preliminary performance analysis are done.

- **SW design and implementation engineering process.** From [ECSS-E-ST-40C], this stage includes elaboration of the software detail design (SDD), detailed design model of the SW components defined during SW architectural design, including static/ dynamic and behavioral aspects. Being a real-time software, UC5 focuses in finding interferences from a co-engineering (CE) method. One example of such source of interference for UC5 is the “shared memory region”. As a possible exclusion mechanism, a new component (called memory scrubber manager) is envisaged, running in both cores, so as every software function is a client of the manager, who allows or not resuming each task.

  The (new) requirements include WCET for the memory scrubbing task, in charge of checking all the memory map in short time (2 minutes). Because the memory is being used by the 2 cores, the scrubber may use the same region of RAM that another task running on the other core is using. IP2/IP3/IP4 analysis may conclude that modification of the architecture is needed, (e.g. by including new safety barrier after IP2 CA) that avoid this kind of collision. Objective is preventing the scrubber function to use regions of memory being already used.

- **SW validation process.** §5.6.4.1 in [ECSS-E-ST-40C] refers to the set of test cases/procedures considering a representative environment, stress/corner conditions, timing test, security specific tests, using "Test/Analysis/Inspection/Design" validation strategy. In AQUAS UC5 formal testing is added as another possible verification strategy; moreover, test/analysis conditions must be multi-concern techniques (in the sense of covering S/S/P in the same testing/analysis). While ECSS standard mentions security-specific testing, performance/stress testing, safety testing (graceful degradation upon safety failures), using representative operational environment. Early validation methods could be applied when following AQUAS methodologies, while according to ECSS, validation must be performed always by "test", unless properly justified.
- **SW verification process.** §5.8.2.1 refers to establishment of the software verification process. In AQUAS UC5, determination of verification effort will consider not only security (but multi-concern instead), verification activities/methods along the PLC of the software might include "early verification" activities.

  §5.8.3.1 refers to verification of requirements baseline. Safety and security analysis (also for performance: ≈ memory/CPU margins) for the requirements baseline is defined as a process based on independent steps, while in UC5 IP1, special attention is put to perform a CA activity with the aim of checking requirements fulfillment feasibility, in search of S/S/P interferences, over defined requirements. Some refinement/modification of the requirements baseline could take place, in case any of the found interferences motivates this (because some requirements enter in conflict among each other and cannot therefore be simultaneously satisfied).

  §5.8.3.3 in ECSS-E-ST-40 refers to verification of the software architectural design: requirements traceability (KPIs), timing synchronization, correct design with respect to requirements and interfaces, including safety, security and other aspects. However, S/S/P verification for the (architectural) design is made sequentially. In AQUAS UC5, IP2, the approach is making an architecture co-refinement and an early validation, taking safety/security requirements and system architecture as inputs. Real-time properties are added to software components in the architecture model, so as to meet performance requirements. As early validation method, schedulability analysis is performed thanks to automatic design tools as CHESS. Modifications to requirements baseline/architecture are proposed and evaluated by the S/S/P experts, in case that performance is not guaranteed. §5.8.3.5 in ECSS-E-ST-40 includes requirements for verification of code. In AQUAS UC5, code early verification, static analysis is done in IP3: the code traced to previous PLC phases (requirements, specs, models). In IP4, WCET validation allows to detect run-time errors (to verify source code robustness).

3.2.1 ESA Guide for Independent Software Verification and Validation

The standards ECSS-Q-ST-80C and ECSS-E-ST-40C both refer to the ESA Guide for Independent Software Verification and Validation (ISVV) [32]. The first version from 2002 as well as the second version from 2008 propose and describe different methods and requirements for verification and validation ([32] Annex F). Each method presented in the ISVV Guide is described as a stand-alone methodology, but a combination of different methods, as proposed by AQUAS, is never mentioned. Since the second version was published in 2008, progress made in the area of V&V since then is not reflected in the guide.

An extensive evaluation of the first version of the ISVV Guide was carried out in 2012 and is described in [33]. The authors examine the usage and usefulness of the guide over ten years, covering 30 projects. The paper points out that the number of discovered issues indicates the importance of ISVV activities and that other domains (e.g. automotive, or even banking) would also benefit from ISVV activities.

However, they also show that the less experienced a development team and the less mature a development process is, the harder is it the execution of ISVV activities.

The guide defines three ISVV levels: ISVVL 0 (No ISVV activities are required), ISVVL 1 (Basic ISVV is required) and ISVVL 2 (Full ISVV is required). The ISVV Level to be applied in a specific project is derived from the Software Criticality Category (SCC) but may be adjusted upward if there are other risk factors warranting increased verification and validation [32, Annex E]. In order to consider S/S/P co-engineering, this granularity of levels is not sufficient and should be increased. Furthermore, there is currently no direct mapping from ISVV levels to applicable methods. For the higher ISVV levels such a mapping should be mandatory and the highest level should rigorously require application of formal methods, including a formal verification of system properties as part of the static analysis.

Formal verification entails a mathematical proof of S/S/P system properties against a model-based system specification, e.g. based on contracts or, more generally, on SysML, MARTE and Object Constraint Language (OCL). This enables the verification of system requirements at system level based solely on the system model, i.e. without the need for the full implementation. For static code analysis with respect to functional contracts, absence of runtime errors or admissible data flows, suitable implementation environments such as SPARK 2014 and FRAMA-C have emerged during the last years. Thus, this contributes to a combined analysis of safety and security properties. Also, transformations between models and source code help to keep design and implementation consistent. A transformation from model to code (e.g. from SysML state charts to C code) could even be automated by corresponding tools. The effort of fulfilling the extensive checklists given in the ISVV Annex G can be significantly reduced. For functional properties an example would be the checklists G.1.5 “Structural Verification” or G.5 “Code Inspection Checklist” which are concerned with issues like “Are the Initial values of the data defined” or “Are all inputs (outputs) of one software unit produced (consumed) by some other unit?”. Fulfilment of various checklist entries could be automatically proven by using e.g. SPARK 2014 or FRAMA-C as programming environments.

For specific safety requirements like WCET the verification may require access to the object code. This is already supported by some AQUAS partners (i.e. AbsInt). Recent methods for WCET analysis like the probabilistic worst case execution time analysis (pWCET [36]) are not reflected in the ISVV Guide. Other time-dependant behaviours like restrictions on event sequences, absence of race conditions or deadlock freedom are not well supported yet.

Runtime Verification (RV) extends the formal verification process to system runtime in cases where the complexity of the system or unpredictable environmental effects prohibit static analysis. RV is based on monitors which allow a dynamic evaluation of predicates regarding system properties. RV, being a recent approach, is completely absent from the ISVV Guide. ISVV actions would not only benefit from RV as a stand-alone methodology but also from the combination of Static Verification and Runtime Verification. For example, as [35] suggests, this could be the provision of counterexamples through generated tests.

[34] presents various challenges with which RV is confronted along with advices like “Monitor specifications should derive from system level requirements and assumptions that have been validated by domain experts.”, “Assure the correctness of the monitor specification.”, “Assured RV should not introduce security vulnerabilities into a system.”, non/low-interference with the monitored system and other properties. In addition to verifying functional properties at runtime, monitors can be leveraged to support more general S/S/P properties like guaranteeing performance indices, detecting security flaws (e.g. intrusion detection) or general anomaly detection. It seems to be desirable that development environments supporting static analysis be extended to support RV for instance by automatically generating monitors. In the optimal case only those parts of system predicates are verified at runtime which could not be verified statically.

3.2.2 Project Management Body of Knowledge (PMBOK)

As a representative example of a framework-oriented standard, the Project Management Body of Knowledge (PMBOK) [56][57] has been selected for gap analysis. The PMBOK is an exceptional and widely recognized "summa" of project management competences, used worldwide to train managers (see *PMI Project Management Institute*). However, there is room for improvement in the area of co-engineering and its sister concepts of multi-discipline, concurrent / simultaneous engineering.

In the current version of the PMBOK, the term "co-engineering" is never found. The term "concurrent engineering" is also never found, nor is "simultaneous engineering". The term "discipline" is used only very generically. The term "speciality" is used occasionally to indicate a specific technical area (such as mechanical or electronic), and discussed in the project organization. But the concept is not further elaborated.

Parallel work is only marginally reported in *Fast Tracking schedule compression* as an approach where "…activities or phases normally done in sequence are performed in parallel for at least a portion of their duration." Clearly, this interpretation of parallel activities does not capture the full semantics of co-engineering.

As a framework-oriented standard, it is reasonable and correct that the PMBOK not focus merely on specific concerns like safety, performance, and security, but on the general concepts of multi-concern / discipline project management – providing exactly the conceptual framework within which co-engineering in the AQUAS sense can thrive. In this sense, we propose the following recommendations for the PMBOK:

Future project managers need to be trained more on organizing and controlling a project through its many disciplines / specialties (e.g. mechanical, optical, electrical, safety, security, performance, energy, waste, etc.). This should become more evident from the WBS (work breakdown structure) and OBS (organization breakdown structure).

The many disciplines should proceed in parallel in a harmonized way, conflicts shall be identified and resolved, trade-off established. The PMBOK may call this "co-engineering" (or another suitably general term), where a pool of experts from the many different disciplines / specialties cooperate on a continuous basis to take major design decisions. Likely, infrastructure facilities may be required to improve co-engineering (e.g. co-presence, co-editing. etc.).

Codifying such a general conceptual basis for co-engineering in the PMBOK could make a real contribution to facilitating its instantiation in the various domain-specific standards.

3.2.3 Arcadia – Engineering Methodology for System, Software and Hardware Architectural Design

This collaborative activity for standards evolution (SE) considers Arcadia from different perspectives and with inputs from TRT, All4Tec, Tecnalia, Magillem and the CEA. It contributes to the Objective 11 of the project, as discussed in section **Errore. L'origine riferimento non è stata trovata.**.

The section begins with an overview of the Arcadia approach and initial planning (first release of this document). It then describes the current status in relation to the AQUAS SE actionable steps described in section 5.2.

The **ARC**hitecture **A**nalysis & **D**esign **I**ntegrated **A**pproach (ARCADIA) emerged initially within Thales in 2007 and is in the process of becoming an industrial standard. It was created in response to the need for a common approach, placing collaboration at the heart of engineering and so reducing the communication barriers between engineering teams, within and across all structured engineering activities of the PLC (see Figure 10). This global method can be customised to specific domains and enables greater traceability assurance, engineering efficiency and mastery of increasing system complexity. The Arcadia methodology is divided into five engineering parts: Operational Analysis Model (OA); System Functional and Non-Functional Needs Analysis Model (SA); Logical Architecture Model (LA); Physical Architecture Model (PA); Product Breakdown (PB). These are all fed by information related to scenarios, data models, data flow, functional chains/operational processes, modes and states.

Discussions for a collaborative analysis of Arcadia by AQUAS partners was commenced early in the project by TRT. At this point Thales Global Services who provide leadership for the Arcadia methodology were contacted about AQUAS plans to provide a change request during the project. The main goal of this collaborative analysis is to provide recommended updates to the Arcadia Methodology with respect to SSP co-engineering. Actions will include a review of how SSP interactions are managed in this methodology (within and across PLC stages), how it relates to the AQUAS methodology, derived recommendations beneficial to Arcadia, type of tooling to support the co-engineering and the submitted change request.

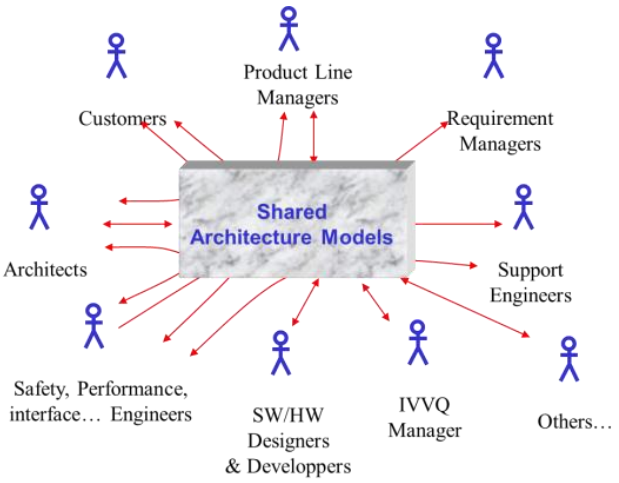

Figure 10: Enhanced connection of stakeholders

Until recently Arcadia was an internal standard within Thales. It was recognised as an official standard XP Z67 – 140 for industry in March 2018, thanks to formalisation with the body AFNOR (the French national organization for standardization).

An integration process is also underway with the International Standards Organisation (ISO), to include Arcadia with the ISO standard approach to Model-Based System and Software Engineering (MBSSE).

There are several steps to achieve the expected work of this task, some of them are in progress and some will be achieved after this report.

Collaborative Activity Status

This part provides a review of the current and planned work in relation to the progress steps defined in AQUAS. AQUAS is being followed by members of the Arcadia committee, a review of the Arcadia methodology has taken place and experience of Arcadia implementation has been provided. Plans on the further contributions have been detailed.

<u>Presentation of AQUAS to the Arcadia Committee</u>

There have been several exchanges with the committee. One of their members has joined our project advisory board and ideally he will be available for the next workshop planned in January 2020.

Investigation (Gap Analysis)

The gap analysis considers the bridging between dependability co-engineering (DCE) and Arcadia. That is, primarily considering the benefits that the AQUAS approach brings to Arcadia and the proposed standard evolution. The study can also raise new challenges to be considered by DCE for future work – and provide supporting references to why the AQUAS/DCE work is so important. Arcadia/AQUAS Communities may also be taken into account.

The most up-to-date description of the Arcadia methodology is available in the book "Model-based System Engineering with the Arcadia Method" [64]. The following paragraphs are related to a review of the book *in relation to dependability co-engineering.* The relevant sections have been quite enlightening and it is presumed reviewing other parts of the book may be complementary on other aspects of the AQUAS approach, such as future use case development.

The importance of co-engineering has prominence in several places within this book covering the Arcadia methodology, this is particularly in Chapter 10 about mixing viewpoints, where it is stated "one of the main difficulties architects encounter is finding the best compromise that meets all the constraints the finalised solution must meet, particularly those known as "non-functional"". Some important information for DCE to take on board for future investigation is that deciding on a trade-off can be a case of both compromises between constraints but also the stakeholders, having "different viewpoints standards, expertise, solution techniques and tools". They also may work to different timeframes. Furthermore, the optimal solution for an individual interdependency may not be the optimal solution for the system as a whole – which means some iteration may be needed to compare the different alternatives. For this reason it is indicated by the methodology that the process should be mechanised and recommends having one reference architecture common between the architectural model and speciality models (but not introducing redundancy).

An interesting example (Pg. 155) considers a feared event for a space launcher and how it links across the architectural perspectives (OA to PBS). Perhaps UC scenarios in future DCE projects should be built up this way. It describes dysfunctional analysis and absence of shared failure modes. There should also be criteria for trade-off or for reconciling viewpoints, making it possible to choose the most acceptable compromise.

Within AQUAS it has been indicated that the coupling of the dependability properties will provide a significant boost for integrating new technologies or managing subcontracted work for products. This is corroborated in Chapter 9, Pg. 144 discussing development contracts – these specify the expectations from a system component so that it is aligned with the overall Physical Architecture and V&V processes. This is critical because normally there is little/no margin to adapt the component once acquired.

As initially introduced, Arcadia splits the architecting of a system up into parts. While AQUAS already has some quite complex use case baseline descriptions on which to develop our approach, these architecting levels would very useful to take into account for future projects. A relevant analysis approach is demonstrated in Section 10.3, considering performance, safety and security attributes across OA, SA, LA, PA, and contracts development.

Chapter 13 discusses articulation between engineering levels. Here the general system development approach uses the "divide and conquer" principle. However, this requires the interfaces between the

sub-systems to be clearly defined. It facilitates the design by reducing the complexity but can lead to system architectures "far from optimal" when there is not a mutual understanding between focuses and conciliation relating to common dependencies. While the chapter focuses on functional aspects of co-engineering, the connotations with separation of concerns of system properties are useful.

There is also a dedicated section on the product line and while out of scope for AQUAS, it will clearly be relevant in future DCE work. While the general push for reusability of components across products drives down the differences, products also need to cater for different market segments (e.g. cars cater for parents driving their children and racing drivers) which changes priorities across the system properties.

This review reveals may relevant areas covered by the Arcadia approach where DCE could be extended in the future for a more complete profiling of system properties to be coupled. It also shows that the AQUAS work is relevant and complementary – where a general approach for coupling system properties and consideration of the product lifecycle will be an important extension to Arcadia. In fact there appears to be a general need in the market for modelling of the PLC with a review of approaches provided by [65].

It can be said that the Arcadia methodology provides a solid approach for system design and an industry proven baseline with which to connect the dependability co-engineering approach currently advanced by AQUAS.

Feedback from usage of an implementation of Arcadia (Capella)

Arcadia proposes a clear and well-defined separation of concerns in systems engineering across the whole PLC. Its model-driven engineering approach (instantiated in the Capella tool) separates traditional layers ranging from the highest level (the definition of a system, operational analysis) to the lowest level (physical architecture). This way, different engineering disciplines (e.g., requirements engineers, functional analysis engineers, architecture engineers, software engineers, hardware engineers etc.) can contribute and collaborate within the same system model. In addition, in the same layer, different diagrams and models are proposed so this is another instance of the separation of concerns principle.

Regarding traceability, all the model elements in the different layers have traceability to other related layers. Propagation links can be used for impact analysis of changes, or to analyse how a given model element has been refined in subsequent layers. This is relevant for co-engineering as, for example, in case of a requirement change, the requirements engineer and the affected engineering disciplines (logical, physical) might need to know the impact and discuss towards a solution.

All the concerns related to the system representation are successfully implemented in Arcadia. Some of the representations can be easily associated to different PLC stages (e.g., operation analysis and the conceptual stage). Arcadia is co-engineering-aware by design, being collaboration and centralized/unified design and development its main strength. For example, co-engineering between the traditional layers (operational analysis to physical architecture) is performed by default. However, co-engineering related to non-functional quality attributes is linked to specialized viewpoints and add-ons. Non-functional attributes which are relevant for a system can be of very different nature (safety, security, performance, energy consumption, mass, price etc.). Some publicly available add-ons are used to capture and analyse non-functional properties separately (e.g.,. performance, mass or price

information[10]), and bridges to external tools can be used to tackle more sophisticated concerns such as safety, security[11], or hardware performance aspects. These tool extensions are usually focused on only one quality attribute, so, combined analysis, interference analysis and other co-engineering helpers, are provided in an ad-hoc or on-demand way. AQUAS provides a conceptual and methodological framework for co-engineering that could be the base for combined analysis tooling within Arcadia and Capella.

ARCADIA & Papyrus

This section discusses compatibilities rather than co-engineering specifics. ARCADIA is a system engineering methodology that aims at improving efficiency and quality, mastering complexity of products, fostering collaborative work, and reducing development costs and schedule. It is divided into steps of operational analysis, system functional and non-functional need, logical architecture, physical architecture, and product breakdown. At each of these steps, it offers a number of recommendations to optimize system engineering. Currently Arcadia is implemented in the Capella tool. In this document we explore the feasibility of an Arcadia implementation in the open-source Papyrus development environment. In particular we study how Papyrus system engineering support can help bootstrap an Arcadia methodology implemented in the tool. In particular, we shall focus on Arcadia requirements for collaborative co-engineering.

Papyrus is an Eclipse-based open-source modeler for OMG-standard UML, SysML, and MARTE. Papyrus also benefits from an eco-system of model-based engineering tools, such as safety and cybersecurity analysis tools, code generators, and Eclipse Modeling Framework (EMF) features. Papyrus in itself does not implement any methodology. Rather it is a tool smith framework to build methodologies with the assets it provides.

From analysing the Arcadia documentation, it becomes clear that several concepts and mechanisms are necessary for a tool implementation of the methodology. The tool should have at least have means to describe **requirements** and manage **traceability** between requirements and between refinements. Means to describe **behaviours** and structural **architectures** is expected. The tool must also allow **separation of concerns**. It should support coexistence of a shared design model with **non-functional analysis models**. Tools for **functional consistency**, and **non-functional validation**, are expected, along with means to support new analysis tools. Finally for the methodology steps themselves, the tool can benefit from a framework to build **automated** methodologies. **Interoperability** between tools and formats is a nice-to-have feature, although not strictly required by an implementation of the Arcadia methodology.

For requirements and traceability engineering, Papyrus has the SysML Requirement stereotype that can be extended for domain and project needs. Requirements and architecture elements can be linked with each other, or among themselves, thanks to the SysML traceability relationships (e.g., satisfaction link, allocation). Requirements, and any their extensions, are supported by Papyrus requirements engineering tools. Such tools ease requirements traceability management (e.g., satisfaction matrix, conformance metrics). For IV&V, test cases can also be modelled in SysML, and therefore traced to requirements.

Papyrus has UML and SysML diagrams to describe use-case, interaction, structured class, state-machine, and activity models. If necessary, the concepts of these models can be extended with the UML profile mechanism. We therefore believe Papyrus has the basic modelling blocks to implement the behaviour description and architecture description means preconized by Arcadia.

[10] https://www.polarsys.org/capella/addons.html

[11] https://www.all4tec.net/safety-architect-capella-bridge

For separation of concerns, Papyrus implements the ISO42020 Architecture Framework standard. In particular the viewpoint concept, in the standard, can be used to separate concerns. Methodology and language designers can define their own viewpoints. The end user can switch between several viewpoints for a same model. Finally, Papyrus has a layer mechanism that can hide or show elements based on the stakeholder's role.

Non-functional annotations can be added to design models thanks to MARTE implemented in Papyrus. Coexistence of shared design model and non-functional analysis models can be achieved through the UML profile mechanism. Indeed, several stereotypes of different UML profiles can be applied on the same UML element, without interference. Furthermore, MARTE supports analysis context modelling in order to explore several solutions based on a same application and platform model.

For functional consistency validation, Papyrus implements the OCL rules defined in UML and SysML. For example, port required/provided interface compatibility is supported by the basic model validation tool of Papyrus. The Papyrus model simulator can also execute models, according to the UML semantics, or an extension of such semantics. Papyrus also has an eco-system of tools dedicated to non-functional analysis. For example Sophia supports safety analysis, with SysML models as input. Papyrus Software Designer supports real-time schedulability analysis with MARTE models as input.

In terms of automation of the methodology steps, Papyrus methodologies are implemented with the Eclipse Cheat Sheet framework. An Eclipse Cheat Sheet offers a step-by-step guide of a methodology and can automate several user commands to perform. In particular Papyrus modelling commands, or command to launch a particular analysis, can be performed by the cheat sheet.

Interoperability is crucial in engineering as a lot of practices are still based on textual documents. As such it is necessary for a performant Arcadia implementation to support tools and formats interoperability. Papyrus has extendable tools for document generation and text generation (including code generation). Such tools can be re-used, and extended, to implement interoperability mechanisms useful for Arcadia.

In conclusion we believe that Papyrus not only supports the modelling languages necessary for an Arcadia implementation, but it also has an eco-system of model-driven engineering tools that would help bootstrap such an implementation.

3.2.4 ANSI/ISA-62443-3-3 (99.03.03)-2013

The ANSI/ISA-62443-3-3 (99.03.03)-2013 ISA99 standard, part of the ISA-62443 series, provides detailed technical control system requirements (SRs) associated with the seven foundational requirements (FRs) described in ISA-62443-1-1 (99.01.01) including definition of the requirements for control system capability security levels, SL C (control system). These requirements would be used by various members of the industrial automation and control system (IACS) community along with the defined zones and conduits for the system under consideration (SuC) while developing the appropriate control system target SL, SL-T (control system), for a specific asset.

Trustport, together with BUT, has elaborated an introduction paper to gaps of the 62443 standard [66].

For Industrial Automation and Control Systems (IACS), the ANSI/ISA 62443 defines procedures for implementing security requirements. The paper introduces the standard gaps analyses, resulting from co-engineering with MTTP over individual AQUAS project use cases (UC1 and UC4), in order to identify some of the missing parts and propose possible extensions for - Security level vector, Impact of security requirements on performance/safety, and Verification methods of requirement implementations.

Based on this analysis the possible recommendations for extending of 62443-3-3 are proposed.

3.2.5 ISO/IEEE11073-00103

The standard ISO/IEEE11073-00103 focuses on communication of personal health devices (PHD). The typical generic personal healthcare use-case described in the standard consists of Agents (typically a software module of PHD) collecting health-data from the user and passing it into a Manager via local short- or medium-range communication interface. The Manager is a software module managing the data and sending it to a consumer: a health-care service such as a medical professional or a fitness trainer. The standard further describes possible intrusion scenarios on each part of this generic architecture (agent, manager, remote system and communication in between).

The first problem of the standard is that, as stated in Section 5.4.1 of 11073-00103, the IEEE PHD standards themselves do not provide methods to ensure security of data exchange with assumption that data exchange is secured by other means. A possible way how to improve on this issue is to extend the standard in a way aligned with the ISO 27001 and 27002 standards of information security management system requirements and the ISO 2799 standard which provides an implementation guideline of the former standards for managing health information security. Another problem of the standard ISO/IEEE11073-00103 is then a missing consideration of safety and performance aspects of PHD.

3.2.6 IP-XACT

IP-XACT is an XML format that defines and describes individual, re-usable electronic circuit designs (individual pieces of intellectual property, or IPs) to facilitate their use in creating integrated circuits (i.e. microchips). IP-XACT was created by the SPIRIT Consortium as a standard to enable automated configuration and integration through tools. Approved as IEEE 1685-2009 in 2009 and published on in 2010.[2] Superseded by IEEE 1685-2014. IEEE 1685-2009 was adopted as IEC 62014-4:2015.

In AQUAS methodology, this standard is used to provide an accurate description of hardware platforms. More precisely, for the Interaction Points, IP-XACT is a key standard as it provides hardware related artefacts usable at several level: concept, design, implementation and V&V.

Here we consider the Interaction Point (IP) descriptions. IP description consist in (1) a collection of concept, design, implementation and V&V artefacts, (2) an analysis document where trade-offs and exploration fields though those artefacts are proposed (3) results of those exploration.

The exploration fields are considering Safety, Security and Performance (SSP) domains.

Thus, in order to facilitate the exploration of studied trade-offs, IP-XACT may be used to store the corresponding values of SSP indicators to be investigated. Indeed, we can use the metamodels of the IP to do this, but it could be convenient to make IP-XACT the handle of such data, in order to facilitate the management of multiple and numerous instances of hardware configurations to be run or simulated.

Proposed extension, localizable at any level of the schema:

Field <IP identifier>

Sub-Field <Security>

Sub-Field <Safety>

Sub-Field <Performance>

3.2.7 ARINC653

ARINC653 is an API specification which is used mainly in the avionics and space domains. The standard specifies the APEX API, which is a set of services together with their behaviour to allow implementing applications, similar to the POSIX API. The focus of ARINC653 is on safety, particularly on partitioning

both time and space, which makes this a standard that is very suitable for implementing mixed criticality systems. Recently, the standard has been extended to include multiprocessor (SMP) considerations, i.e., the ARINC653 threading model has been extended accordingly.

An API specification standard may not be the first that comes to mind when trying to find ways to promote a safety/security/performance co-engineering method, because this standard is not about methods or processes. But APIs are always specified with a given mindset that has implicit considerations that may contribute the processes within which the APIs are used – e.g. since the focus is on avionics safety critical system, for sure the corresponding certification standard where well known to the authors of the standard. So the ARINC653 standard was chosen here to examine how well the security and performance aspects are considered or may be considered by extensions to the standard.

3.2.7.1 Security

The ARINC653 standard is mainly driven by safety considerations, e.g., it allows to partition CPU time and memory space so that mixed criticality systems can be easily implemented. This partitioning is also a good fit for basic security demands: the separation of time and space is necessary for isolating a system from a potential attack.

The ARINC653 standard, however, considers no specific security needs that could go against safety, e.g. when encryption algorithms become necessary to ensure secure communication, this introduces complexity that is a potential threat to safety. There are no considerations how this can or should be handled, and there are no APIs that directly support cryptographic routines. These considerations need to be done at application level and at system design level when using APEX. It would be conceivable to extend the standard by security consideration for every API service, e.g., for message-based communication, it would be possible to propose ways to communicate securely and how this could impact safety. Even API extensions like signature checking would be possible.

Also, some APIs have covert channels in some implementations that may not be obvious. This is currently not explicitly considered, either, because it is irrelevant for safety. One example is a potential covert reverse channel in queuing ports: these message queues are unidirectional, so there is a reader and a writer, which may be in different partitions, and, thus may be differently critical. The queuing port counts the number of messages in the port, and this can be read from both ends of the communication port. In some implementations that tightly couple the sending and the receiving ports of a channel, this may be usable by the reader to send bits back to the writer by consuming or not consuming messages. I.e., the state of the consumption could be visible to the writer, so security cautious implementation could be warned to avoid this, and application developers could be warned to check the implementation if they need to know about such covert channels.

In an SMP system, there should also be some considerations about running time critical threads from different partitions at the same time on different cores. Modern hardware always has shared resources, and while this is well-known today, some considerations about this could be included in the API descriptions so that this becomes obvious to users of the API to avoid false assumptions.

3.2.7.2 Performance

The ARINC standard seldom explicitly considers performance in the API specification, e.g., there is no expected complexity annotation for the API, while in many cases, programmers will have some intuitive idea, e.g., that message sending is expected to take O(n) where n is the length of the message. This is not specified, however, but could probably easily be added to where there is common understanding of the expected complexity anyway. For some API calls, it is less obvious and may also help, like how complex the creation of ports and threads is and/or whether ARINC653 has any restrictions here or not.

In some cases, the ARINC design is driven very much by only functional ideas, but still, maybe with having an implementation implicitly in mind. This may cause the ARINC standard to 'accidentally' complicate the ability of implementations to be more efficient. One example is the definition of queuing port lists, which is an API to block for message events on multiple ports. The idea of a 'list' is obvious from the name of the structure, but the consequences of this assumption are not explicitly encoded in the description of the behaviour. E.g., in the API definition, which port will be 'tested' for being ready to receive/send next follows the order of the list. However, the expected complexity is not specified: but searching the next empty item in a list is probably O(n), not O(1) and not O(log n). This is (a) simply lack of information, and (b) an accidental read block for more efficient implementations, because the 'list' view may not allow the use of faster data structures internally. It would be a sensible extension to add the ideas of the complexity explicitly, and to also think about whether the implicit ideas block more efficient implementations, in this case maybe the standard could be more cautious about the definition of the API behaviour by specifying only the behaviour that is really required, so that nothing is implied about a specific implementation detail, like the underlying data structure.

3.2.7.3 Co-Engineering

The previous sections have already listed a few things that have the potential of adding information about security and performance to the ARINC653 standard. To avoid overhead, the question remains whether there is a way to find single points of interactions where in an API standard, the three aspects can be considered together, so that in most places, support comes naturally without further joint consideration. For an API specification that defines services, this problem is solved almost naturally: the overhead of a continuous combined analysis has a natural interaction with the API calls themselves. I.e., each service, which is already specified with its scheduling behaviour, granularity, etc., for safety, could also specify the security and performance aspects like pre- and post-conditions at the API level. This way, the three aspects interact 'inside' of the service, while to the outside, they provide a clear pre- and post-condition specification the user of the API could rely on. The application built upon this could then also use the API level as one point of considering the aspects together, and separate analyses can be performed between the API calls.

3.2.7.4 Summary

The ARINC653 standard has a good potential to be extended to security and performance considerations by defining an API level that provides a natural layer where the three aspects can be considered. It can also be extended gradually, and improved gradually, which would make it easy to boot-strap the potential improvements. The gain would be an early awareness of security and performance together with the safety aspect that is already present, among application developers. For system integrations, the natural interaction layer would have the advantage of providing structure for extending the overall safety/security/performance strategy.

SYSGO is part of the ARINC653 committee.

3.2.8 VEL by OASIS

The OASIS Variability Exchange Language (VEL) is an upcoming standard at the time of this writing for exchanging variability information across variant management tools and systems development tools [27]. As represented in the following figure, the current variability information consists of the defined variation points within the system, and the established configurations of the variants.

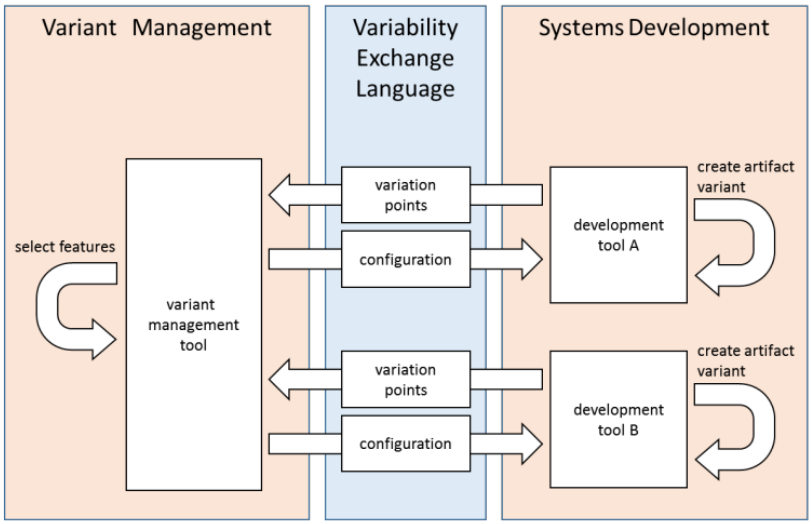

Figure 11: Variability Exchange Language and its relationship with variant management and systems development [28]

Being able to express variability is a key technical mechanism for the evaluation of design or implementation alternatives. In the AQUAS project, we emphasize and evidence that combined analyses of different quality attributes of the system can trigger the need to take trade-off decisions and the need to evaluate different design alternatives. These evaluations of alternatives have then a tight relation with co-engineering given that design decisions are not always under the competence of a single engineering discipline.

VEL was conceived to express how assets (agnostic of the type of assets) can vary based on variation points. Variant management is at the core of engineering families of systems to produce and maintain variants according to individual clients' needs. The next figure shows an excerpt of the content of a VEL file [29]. We can find two mutual exclusive alternatives (*xor-structural-variationpoint*) in a physical architecture model (corresponding to the needs of two different companies). Here they defined the model elements (identified by their *uuid*) associated to each alternative.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<variability-exchange-models id="root">
    <version>0</version>
    <capability>...</capability>
    <variability-exchange-model type="variationpoint-description" id="Physical Architecture">
        <xor-structural-variationpoint id=ACU">
            <variation id="Albonair">
                <corresponding-variable-artifact-element>
                    <uuid>de74b43c-4c43-4506-9c61-a210d79cdfe1</uuid>
                    <uuid>c9551405-09f1-4274-a822-3d95f6d44f3f</uuid>
                    <uuid>29e2baef-4d0a-40fd-9bbb-f65331c4d3f6</uuid>
                </corresponding-variable-artifact-element>
                <condition type="single-feature-condition">Albonnair</condition>
            </variation>
            <variation id="Bosch">
                <corresponding-variable-artifact-element>
                    <uuid>455e2c8c-391e-4f84-84fa-35e2883b41a6</uuid>
                    <uuid>31efc977-05c5-4f4f-8179-1c6dd78f0ba2</uuid>
                    <uuid>c7601d85-733a-463b-a1f1-f85ab3d4d170</uuid>
                    <uuid>aace2ef1-3ef0-48a6-82fa-df57f8075d13</uuid>
                </corresponding-variable-artifact-element>
                <condition type="single-feature-condition">Bosch</condition>
            </variation>
        </xor-structural-variationpoint>
    </variability-exchange-model>
</variability-exchange-models>
```

Figure 12: Example of a VEL file [29]

Beyond its obvious use for system families and product lines (the main intention of VEL) [29][30], variability-awareness can also be crucial for other engineering use cases such as the development of adaptive systems, or in the exploration of alternatives in single systems (e.g., design, architecture and implementation) with respect to possible trade-offs among quality attributes (e.g., safety, security and performance). The latter case is of high relevance in the AQUAS context.

Functional requirements are usually discussed based on customer/market needs, but trade-offs on decisions regarding trade-offs on non-functional requirements are more difficult to made. Traditional examples are trade-offs among safety, security and performance [31][32][33]. However, this is a per case/domain problem as the non-functional qualities under investigation could be also maintainability, reusability, power/energy consumption, testability, latency, robustness, reliability, usability, cost, size etc.

In this domain, the most critical step is related to the Design Space Exploration activities [34] and the main differences among the works in the literature are mainly related to the different amount of information and actions that explicitly rely on the engineer experience.

Trade-off decisions among quality attributes (which can be mapped to the multi-criteria optimization problem) requires evaluations of alternatives in the design space. Thus, to analyse alternatives, a design space exploration setting needs to be established (i.e., a way to define how the alternatives vary and a method to evaluate each variant).

A multicriteria optimization problem [35] can be defined as the problem of finding a vector of alternatives which satisfies constraints and optimizes a function whose elements represent the objectives into a search space of alternatives. These functions represent a mathematical description of performance criteria which are usually in conflict with each other (i.e., orthogonal). Hence, the term optimize means finding a solution in that search space which would give the values of all the objective functions acceptable to the input requirements.

The search space of alternatives can experiment a combinatorial explosion making it impossible to evaluate all variants [36], so methods to effectively explore or analyse the space has been proposed [37].

Regarding technical barriers, instantiating a design space exploration setting is complex and cost-effective solutions are difficult to put in practice. There are approaches ranging from manual [38],

semi-automatic [39] or automatic [40], and usually with custom tools and frameworks [41]. Moreover, automatic solutions usually fail to scale to complex/large systems [42]. Time to market is also relevant and given that iterating over possibilities is time and resource consuming, at some point, it may compensate to proceed to the next stage even if the current solution could be improved (i.e., an optimum was not reached, or a few criteria/requirements were still not satisfied). This might be solved in further iterations incurring in acceptable technical debt.

A common language to express alternatives could surely help in the design space exploration domain. The state-of-the-practice might be improved by adopting standards where variability information can be exchanged between tools and tool features can exploit the information to create variants. There is no gap on using VEL regarding the creation of variants. However, creating variants is not the only important aspect of design space exploration given that information regarding the impact of variation points and configurations in quality attributes is also necessary to evaluate the alternatives.

In the software product lines domain, several approaches to add quality-related information has been studied [43]. One of the most widely accepted ways are the extended feature models [44] which are variability models with quality attribute annotations. Several generic tools have been proposed such as SPL Conqueror [45], ClaferMoo [47] (with visualisation support for humans to analyse the possible solutions [48]), MO-DAGAME [49], SATIBEA [50] or SPL Config [51]. Apart from these tools, other specialized tools have been proposed such as TTool for safety, security and performance analysis in design spaces [52].

Existing works use to rely on the creation of an extra asset, the extended feature models as mentioned before, to express how a variation point impacts non-functional qualities. In the case of design space exploration, VEL should be enough and more practical to capture this information so ways to include quality attribute annotations within VEL constructs is desirable.

We envision two main usage scenarios:

1. Tool features might be able to use this information in the VEL document for helping (automatically, semi-automatically) in **selecting the variation points** which are more appropriately based on non-functional criteria.
2. **Tools providing analysis of the variation points** can feed and consolidate information on quality attributes to the VEL document. These tools are usually specific to one kind of objective (quality attribute) so the VEL document can gather all the results.

As a simple example of usage scenario 1, we might have a system where the total mass of the system is a relevant non-functional property (e.g., an unmanned aerial vehicle). The system design is modelled in Polarsys Capella using the mass add-on [12] so components have an associated mass. The system includes variability so depending on the customer needs, the selection of features (and therefore components) varies but there is a strong constraint of the maximal mass with a constant threshold that cannot be exceed. The information on the mass and of the variation points are in the development tool, however VEL does not define a standard way to exchange this information on the mass non-functional property to the variant management tool. Similarly, it can be the case of performance[13]: Having or not an optional component might affect the time-consumption of a functional chain with safety requirements on the expected performance.

As an example of usage scenario 2, the design space can be explored to identify how features and feature interactions affect the targeted non-functional feature (e.g., performance). Tools such as SPL Conqueror [45] allows to measure non-functional properties after the software product line is developed and thus the variation points are already defined. In this case, the measures on how

[12] https://wiki.polarsys.org/Capella/Viewpoints/BasicMass

[13] https://wiki.polarsys.org/Capella/Viewpoints/BasicPerformance

variation points affect the non-functional properties can be stored in the VEL document to be exploited by the variant management tool.

In an available version of the VEL schema [14] the class "SpecialData" is the construct enabling the addition of application specific information to variation points and variation objects. This is based on a map of key value pairs which is like a plain-text properties file. While this bring freedom to add any kind of information, the non-functional properties aspect could be more formalized with a dedicated construct. This will require to modify the VEL schema to be able to have dedicated constructs to add this information on variation points. The objective is a standardized way within VEL to exchange information which can be directly exploitable by the optimization capabilities of variability management tools. Without this, the standardized "SpecialData" information could not be parsed in a generic way by these tools. It is possible that constructs to list the relevant quality attributes, how the quantities are aggregated for the different variation points, and their fitness function (maximize, minimize, others) might be interesting. As a remark, acknowledged issues in measuring the impact of variation points in quality attributes are the feature interactions or the interactions of variation points. This should be also considered when defining ways to express variation points impact on quality attributes.

As suggested by VEL experts that we contacted (the VEL chairs), using the parametric variability capabilities of VEL can be an alternative to the use of "SpecialData". "ParameterVariationPoint" is a kind of variation point which is used to set values in the targeted system (e.g., a constant String or Int in a source code file or model attribute). However, even if its semantics were not for this objective, it can be also used for the purpose of, not only defining values on non-functional features, but also defining dependencies with other variation points using the existing VEL dependencies and constraints mechanisms. A similar approach has been also used in feature models where non-functional features (without an assets/implementation counterpart) are added just to express dependencies and constraints to functional features or other non-functional features (e.g., [46]).

We received feedback on our gap analysis from the VEL chairs of this OASIS initiative. Even though they considered it an interesting extension, at this stage, the only current priority in short and mid-term is on the VEL core functionality, restructuring the schema to make it more general and to allow the safe introduction of user extensions (i.e., not breaking the schema or introducing incompatibilities). Logically, in-depth discussions on extensions at this stage can cause delays on the standardization of the core. They also pointed out that an extension on this direction should be preferably based on standard ways and formalized representation of quality attributes. Therefore, the gap on the non-widely-known solutions on a standard for representing quality attributes, their values and metrics, as well as their dependencies and constraints, is also hindering an initiative for this potential future VEL extension.

3.2.9 IEC 61508 and IEC 63187 – Safety and Cybersecurity, current status of discussions

The basic functional safety standard IEC 61508 is currently under revision, trying to take into account evolving technologies (mainly in the software part, but also in the system part, particularly from system engineering). IEC 61508:2010 was the first functional safety standard linking to security impact on safety, setting requirements on the need to consider possible security threat impact on safety during risk and hazard analysis by including in this case a threat-risk analysis.
"If the hazard analysis identifies that malevolent or unauthorised action, constituting a security threat, as being reasonably foreseeable, then a security threats analysis should be carried out."

[14] https://www.variability-exchange-language.org/

If this is the case, in the following processes the security issues have to be managed over the whole life cycle, e.g. by the safety manual (mandatory requirement).

For Ed. 3.0, it was envisaged that the interaction points or links between safety and security processes should be outlined more detailed, e.g. similar to other derived domain standards as mentioned in this document (e.g. as in ISO 26262 with references in several parts, additional mandatory requirements and a guideline for the interaction between the safety and security teams, e.g. as explained already in the IEC TR 63069 under “co-engineering”.

There are now three proposals on the table for further discussion in the Vienna Meeting of IEC Maintenance teams MT61508-1/2 and MT 61508-3 in December 2019, but before to be commented by the national Committees until October 31st 2019. The comments should be handled by the Joint Task Group JTG 06, Cybersecurity before December. This path was taken, because it was not possible to achieve consensus between the groups. All groups are convinced that the cybersecurity impact on safety has to be considered, but there is no consensus how far. One group insists that it cannot be scope of a functional safety standard to set up mandatory requirements, but has only to reference the need for interaction and the IACS 62443 standard (IACS security). The other two proposals request several mandatory requirements and the need for further guidance on interaction and/or co-engineering. AIT has always proposed and supported the more detailed and rigid approaches to include co-engineering aspects, but there have been even emotional discussions on these topics in the past. AIT has submitted comments on these issues focussing on the co-engineering aspect.

A new work item, IEC 63187, Functional safety - Framework for safety critical E/E/PE systems for defence industry applications, has been started as a daughter standard of IEC 61508. One of the goals of this standard is to fill some identified gaps of the current version of IEC 61508:2010, which is rather not a system engineering standard, and does look on functional safety controls rather than on “systems of interest” in a holistic context of “system-of-systems”. This includes the dependability properties of these “systems of interest”, particularly safety and cybersecurity, but also on others in the future. The work was started about a year ago and provides another opportunity to include AQUAS co-engineering aspects.

3.2.10 Requirements for MARTE evolution

In this section we provide a synthesis of requirements expressed by AQUAS partners that are relevant for the project:

Explicit support for security is needed. A few ideas follow:

- Allow the description of security requirements, with a specialization on: confidentiality, availability, authenticity, etc.
- Allow the description of attack trees, with (of course) an explicit <<Attack>> stereotype
- Handling basic security mechanism like:
  - Encrypt/decrypt, asymencrypt/asymdecrypt, hash, signatures
  - Use of an extension of VSL to describe security mechanisms as security functions
- Support for handling well-known security protocols like:
  - Diffie-Hellman, TLS
  - Specific data ports (maybe as a security tag on data ports)
- Explicit tag/stereotype/value for security data like:
  - Public/private keys, certificates
  - Shared elements between classes, e.g. for being able to indicate that two keys are the same in two different classes/blocks when the system starts

- Description of the security of architectural elements
  - e.g., privacy of a bus, a memory
  - Possible specific attributes/values
- Allocation of security mechanisms and data
  - Sec. mechanisms allocation to e.g. execution nodes
  - Data (e.g. keys) allocation to memories
- Attacker view vs. system view
  - Explicit stereotypes/diagrams/views for capturing attackers' capabilities
  - Be able to link attacker capabilities to countermeasures, e.g., security mechanisms
  - Allocation of attacker capabilities to the architecture, e.g., allocate a spy capability to a bus

In the context of Co-Engineering:

- It may be useful the explicit modeling of interference between requirements (safety vs. security, or more generally between requirements whatever their nature)
- Eventually considering an «interfer» relation
- Also the explicit relation between timing issues and security constraints (maybe in VSL)
- It may result convenient adding an attack tree diagram/view, Attack fault tree, and interference between the two

- Support for contract-based design
  - Contract-based design is a well-known practice for system specification
    - supports stepwise refinement, compositional reasoning
    - supports component reuse
      - evidence/argument fragments reuse (e.g. AMASS)
    - static and run-time usage
      - support at run-time for open collaborative systems (e.g. SafeCOP)
  - Currently MARTE has limited support for contract specification (NfpConstraint with kind=contract)
    - Extended from Contrex proposal
  - Needs
    - Be able to discern between assumption and guarantee properties, e.g. to enable finer-level traceability to requirements, different criticalities allocation
    - Support for strong and weak contracts concepts
      - A component having a strong contract associated cannot operate in a scenario where the assumption is not fulfilled by the environment

- A component having a weak contract associated can operate in a scenario where the assumption is not fulfilled, but its guarantee is discarded

- Support for contracts decomposition, as for requirements
- Support for contracts-based design at component-classifier and instance level, e.g.:
  - Be able to associate a set of weak contracts to a component-classifier
  - Instantiate the component-classifier in a given context and select the weak contracts which hold

- Support for dependability information
  - No support in MARTE
    - Some support available from OMG Dependability Assurance Framework for Safety Sensitive Consumer Devices RFI
  - Needs
    - Modelling of fault-error-failure (states) for a given component
    - Modelling of failure propagation between components
    - To allow quantitative/qualitative dependability analysis
- In order to support system level modelling, in a way that results are compliant with IP-XACT IEEE1685, standardized semantics and transformations from MARTE elements to IP-XACT are needed.

3.3 Current approaches being pursued by standards developing organisations

In the following, some examples of co-engineering approaches being pursued by SDOs are given (not necessarily in the specific domains of the AQUAS use cases). They are primarily concerned only with safety and security.

3.3.1 IEC TC 45, SC45A - Nuclear Power Plants

The series of nuclear power plants safety and associated cybersecurity standards are a good example of a very good separation of concerns in the documents, and, on the other hand, integration of co-engineering aspects by a coordinated approach. They chose a three-step approach:

- **IEC 61513** (*Nuclear power plants. Instrumentation and control important to safety. General requirements for systems*): focusing on safety (Nuclear Safety domain standard interpreting IEC 61508)
- **IEC 62589** *(Nuclear power plants - Instrumentation and control systems - Requirements for coordinating safety and cybersecurity):* setting up “fundamental principles” to protect the safety objectives despite cybersecurity threats, avoiding adverse impact of cybersecurity counter measures. This represents the “safety first” viewpoint, view on security measures from the safety point of view.

Some of the "fundamental principles" are (examples, excerpt):

5.2 Fundamental Principles

- **Cybersecurity** shall **not interfere** with the **safety objectives** of the plant and shall **protect their realisation**. It shall not compromise the efficiency of the diversity and defence-in depth features…
- **Cybersecurity requirements** impacting the overall I&C architecture shall be **addressed**…
- Implementation of **cybersecurity features** shall **not adversely impact** the required performance (including response time), effectiveness, reliability or operation **of functions** important to **safety**.
- The **failure modes and consequences** of cybersecurity features on the functions important to safety shall be **analysed** and **considered**.
- Any **architectural property or characteristics** initially designed for safety reason (e.g., independence between systems), and later considered as a potential cybersecurity counter-measure … should be re-examined on purpose … to confirm its cybersecurity added-value

- **New work item IEC 63096:** "*Nuclear power plants - Instrumentation and control systems - Security controls*", specifically focusing on the selection and application of computer security controls from the included security controls catalogue" (based on IEC 62645, top level document for cyber security, and IEC 61513), which has now reached the CDV status (CD for Voting), the publication is planned October 2020:
  - To ensure consistent understanding of the process of the selection and application of cyber security controls;
  - To ensure consistent understanding of which security controls are recommended and optional for the security baseline and the security degrees S1, S2 and S3 (Catalogue)
  - to describe a method for crediting/inheriting existing security controls and safety provision for I&C systems important for safety
  - To describe a method for applying compensatory security controls in case recommended security controls cannot be implemented
  - To describe a method for handling of the legacy topic.

3.3.2 IEC TC 44, Safety of Machinery – Electro-technical aspects

With respect to the general domain of machinery safety, the following activities are observed:

- **IEC 60261** *("Safety of machinery: Functional safety of electrical, electronic and programmable electronic control systems"),* is the domain standard interpreting IEC 61508 (plus complementary ISO 13849-1 for other safety aspects than E/E), for security of IACS (Industrial Automation and Control Systems) is IEC 62443 the basic standard.

- **Cybersecurity:** Originally, the idea was to separate safety of machinery (responsibility of the manufacturer of the machine) and cybersecurity responsibility (on the shoulders of the OEM/integrator). This early concept was rejected by the IEC ACOS (Advisory Committee on Safety) as well as by ISA 99 (International Society for Automation, before known as "Instrument Society of America") with the argument, that a separation of requirements cannot be at this low level. (Informal comment: "Just a firewall is not sufficient!").

- **New work item has been** started: *"Security aspects related to functional safety of safety-related control systems"*: now a much better approach "to consider the security aspects in context of safety of machinery", a similar approach as the second level standard of IEC TC45 SC45A, Nuclear Plants)

The following aspects will be considered:

- what is the relationship between safety and security?
- vulnerabilities can be the result of systematic faults which can lead to a hazardous situation of the machine;
- vulnerabilities may impact the integrity and availability of the safety-related control system to properly perform its function(s);
- reasonably foreseeable misuse (see ISO 12100), e.g. typical use case definition and application of a corresponding threat model.

4 AQUAS influencing standards

4.1 Overall approaches to influencing standards

It is well known that standards tend to have a long evolutionary lifecycle – not merely because of "inertia", but also because an essential characteristic of a standard is the need for a sufficient period of guaranteed stability to fulfil its role as a point of reference and compliance within its community of users. On the other end committee meetings are quite strictly organised and focused. The AQUAS consortium is well aware that it will not always be possible to synchronize its standards-influencing efforts with the windows and opportunity that will arise during the evolutionary cycles of the standards developing organisations.

Nevertheless, activities can be engaged that are useful even in the absence of perfect synchronisation with the standards renewal cycles. The following approaches are considered in AQUAS:

- **Reports and change request packages**. Even in the cases where the SDOs are not currently in a public consultation or updating cycle, a package can still be prepared containing reports and/or change requests that may be presented at the next possible updating cycle of a standard, also beyond the end of the project. This is particularly viable when a member of the AQUAS consortium is a member of the relevant standards committee and can take direct responsibility for presentation of the change requests at the opportune moment. In a sense, this may be considered a type of "leaving a legacy for posterity".
- **Presentations to standards committees and working groups**. Even outside of updating cycles, standards working groups are often active – for example, to collect experience reports and suggestions from users for future revisions of the standard (which may still be years away). It is common usage to present new ideas and technologies to these working groups to inform them of recent developments to take into consideration. AQUAS members can prepare targeted informative presentations for such working groups. Indeed, this has been formalized in the awareness-raising objective of AQUAS.
- **Guidelines**. It has become common practice to prepare guidelines for the usage and interpretation of standards in particular ways, especially when the standard is unlikely to be updated for several years. In this way, an informal type of *ad hoc* modification to the standard is achieved. For example, a guideline entitled "Co-engineering with Standard X" could provide a set of recommended best practices that augment the standard with facilities for achieving co-engineering (e.g. additional intermediate steps), but remain conformant with the text of the standard in its current form.
- **Dissemination of gap analyses.** Publication and presentations to international conferences of co-engineering gap analysis for representative standards, that are not reachable in the project time frame by members of the AQUAS consortium, should raise in the long term the awareness of the challenges faced with the provision of safe, secure, and efficient cyber-physical systems and the needs for evolution.

4.2 Report on the Evolution of Co-Engineering Standards

The following sections summarize the final status of the Standard Evolution activities (extending what reported at M30 in D1.9 v1.0), in relation with the specific SE4CE objectives of the project.

4.2.1 Objective 9: Change Requests

Contribute to the improvement of standards to address co-engineering, by submission of change requests to at least 1 standard for each of the AQUAS use case domains.

The use cases provide the most immediate potential opportunities to submit change requests, because AQUAS partners are adequately positioned in a number of cases.

ATM Use Case

ISO TC20/SC16 (TC20 – Aircraft and Space Vehicles, SC16 Unmanned Aircraft Systems (UAS)) recently started standardization in the field of UAS including, but not limited to, classification, design, manufacture, operation (including maintenance) and safety management of UAS operations (UTM – UAS Traffic Management). This committee has at the moment only standards under development which means we have an 'Open Window of Opportunity':

ARINC-653 provides a baseline development environment for UAV applications used within an Integrated Modular Architecture. Basically, this standard allows UC1 to incorporate space and time partitioning using a safety-critical avionics real-time operating system (as is the case of SYSGO's PikeOS), enabling that different applications (e.g. flight controller system or payload applications) to be executed independently from each other without interference in terms of CPU processing and memory resources. Taking advantage of the participation of SYSGO within the ARINC-653 committee, we plan to provide our progress and results to this committee in order to be considered for future updates or extensions.

Medical Use Case

In the medical use case, there are three potential improvements to four standards improvements (IEEE 11073, EN62304, EN-60601-1-10 and EN 62366).

EN 60601-1-6, EN 60601-1-8, EN 60601-1-10, EN 62366-1 and EN 62366-2 have been reviewed by City. AIT and RGB for its evolution. ISO/IEEE11073-00103 has been analysed by Trustport and BUT.

Partner RGB has provided contact information of the leader of a medical cybersecurity group, which is a potential point of influence for AQUAS.

CITY also had contact with TUV Sud. They are involved in evolving security standards for medical devices and expressed interest in the AQUAS's relevant standard-related activities. A visit to CITY was planned at the end of 2019 to explain the AQUAS co-engineering concept and observations on standards, delayed first due to University strikes and finally for the Covid-19 emergency.

Rail Carriage Mechanisms Use Case

There are new developments with regard to the new generation of trains, e.g. the European Railway Traffic Management System. For example, in **CENELEC TC9X,** work is going on with respect to integration of security considerations in the existing standards landscape. DIN VDE V 0831-104 "Electric signalling systems for railways – Part 104: IT Security Guideline based on IEC 62443" was a "forerunner" in proposing to include the lower SL (Security level 1) of IEC 62443-1 in the safety standards. AQUAS partner AIT has made contacts with SC9X SGA16, which recommended to consider security in the CLC TC9X standards' framework. The chairperson, together with AIT and others, organized the so-called "Safety meets Security" workshops of "Hanser Tagungen", where the co-engineering aspects are disseminated. Therefore, there may be some impact in the longer term of evolution of AQUAS standardization work.

In the meantime, **CLC/prTS 50701, "Railway Applications – Cybersecurity"**, was released as a draft mid2019, which will make available a Cybersecurity standard that covers not just Signalling, Rolling Stock, or Fixed Installations, but the whole Railway System.

One objective of this standard was to allow and enable the railway domain to utilize secure components from other domains, and therefore compatibility with standards from the series of IEC 62443-3 was a requirement. This allows to utilize components developed after IEC 62443-3-X and EN 50701 will here replace IEC 62443-4-X with railway specific guidance on the system level.

The goal was to establish a TS which defines a unified way how to handle and address cybersecurity over the whole railway sector, which is based on existing security standards (IEC 62443). The system lifecycle will be based on EN 50126 and therefore extend RAMS, which always included unintended negative actions and required therefore already access control with a full view on security. It will define a detailed risk analysis process for railways and adapt the system requirements from IEC 62443-3-3 to the railway domain.

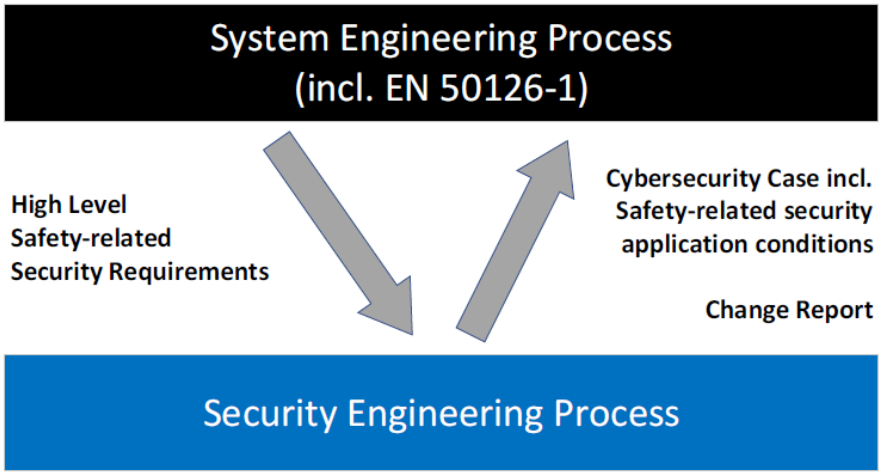

Figure 13: Security in Railway

An important achievement will be the methodology on how to handle safety and security, this is planned by connecting the safety case with a security case. This interaction is done by identifying high level safety-related security requirements in the system engineering process. The cybersecurity case has then the task to demonstrate if these high-level requirements are fulfilled, are fulfilled under the assumption that some Safety-related Application Conditions (SRAC) are considered or are not yet fulfilled and compensating countermeasures have to be defined.

Based on this assurance processes and requirements will be defined. In addition, it will extend the existing requirements and recommendations (e.g. the hazard log) with guidance on vulnerability management.

Industrial Drive Use Case

The Industrial Drive use case involves IEC 61508, as discussed earlier, where partner AIT is involved, and also AMT, as well as a set of standards around IEC 62443. Discussion on these standards is reported in Section **Errore. L'origine riferimento non è stata trovata.**, as they are also relevant for Objective 10.

The UC4 example, provided by SIEMENS, will be used by MDS when presenting the IP-XACT extensions to IEC 62014 (s. Chapter 3.2.6).

Space Multicore Use Case

Several ECSS standards are relevant to **Space Multicore Architectures**, including E10, E40, Q80, Q30, Q40. Recently the ECSS organization has promoted a new E40 WG to integrate security requirements from the ESSB-ST-E-008 standard requirements within changes into E40.

More numerous ESA Programmes, in which the EU/EC is involved, e.g. Galileo, GMES, SSA, have stringent security requirements, mainly due to their safety and/or civilian/military dual aspects. This imposes higher consideration for secure software engineering than ever before. These considerations are now gathered into an ESA internal document, which describes the impact on E40 and Q80.

Updating these Standards to reflect security will be useful to mainly secure ground software, but not only.

INT participated in the past to the ECSS Software Standards Working Groups for E40C, E40-01 (Software Engineering Handbook), E40-01A (Agile Software Engineering Handbook) and Q80C.

In March 2019 INT contacted the convenor of the E40C WG to ask for any update on the ECSS related to any extension to introduce system and software level security requirements in the ECSS and to address co-engineering of SSP. As a matter of fact, 3 months later the ECSS Technical Authority has decided to start a new revision of the E40C with main purpose to bringing the ESA Internal Secure SW Engineering standard (reference ESSB-HB-E-008-Issue1 (4July2016)) to ECSS level.

INT participates to the new E40 working group and will be able to promote awareness on co-engineering and elements of the gap analysis performed in AQUAS. The WG kick off meeting was held on 1st October 2019. However committee meetings are strictly organized and focused on already existing Change Requests produced by the ECSS members (essentially the European Space Agency, European national space agencies and EUROSPACE). The ESSB standard is poor, addressing SW only, while security needs to be addressed starting from the System level in the ECSS, as for safety and dependability. AQUAS Specific Change Requests will also be allowed once the ECSS-E-ST-40 revision will go on public review.

Cross-Domain Change Requests and contributions during standardization WG meetings

AIT is active in several cross-domain (or: domain-independent, generic) standards, where change requests or direct contributions during working group meetings have been submitted or are currently under submission, always with the focus on safety and cybersecurity co-engineering, to keep the standardization landscape as conformant as possible in this respect. It should be noted, that the process is not just defined by or limited to a single submission of a change request – for some time (sometimes one or two years) ideas are discussed in the working groups based on recent comments of national standardization committees, and on contributions of members (experts) present in the meetings. This was and is done by AIT in the Maintenance Groups of **MT 61508-1/2** and **MT 61508-3** of IEC SC65A, in IEC TC 65 WG 20 (“Framework for functional safety and security”, **IEC 63069**), IEC SC65A WG 17 (Human factors and functional safety), IEC SC65A WG 18 (Functional safety standards for defence industries, which has the major goal to bring IEC 61508 further forward towards a system engineering standard). Work for ISO TC22, Road vehicles, standardization groups is worth mentioning, although AQUAS has no automotive use case, but is following the same concepts of co-engineering in this domain, as well as cybersecurity in Smart Manufacturing and Robotics (see previous sections).

In IEC 61508, working on a CD planned begin of 2021, AIT submitted/supported change requests on safety & cybersecurity engineering (for the system part 1 and software part 3), supporting a strong representation of requirements for co-engineering, and change requests concerning an integrate approach to consider more human factors. Since consensus is needed, and as there have been strong objections to include too detailed guidance on implementation, a compromise was achieved December 17, 2019, at the Vienna meeting. The most important achievement is the fact that an explaining statement is now in the Scope and a mandatory requirement is now in Part 1 (System):

- Add text to Scope of the standard at the end of clause 1.1:

  “The scope of this standard is the achievement of functional safety for E/E/PE safety-related but, apart from normative requirements in the hazard and risk analysis phase, does not itself provide normative requirements for malevolent action arising from a cyber security risk. However, if a cyber security assessment has identified that a reasonably foreseeable cyber security risk will arise, it is essential that measures be taken for all relevant phases of the Overall, E/E/PE and Software Safety Lifecycles in order to protect against such threats to ensure that functional safety is achieved.

Note: For requirements and guidance on cyber security see e.g. IEC TR63069, IEC 62443 series and ISO/IEC 27000 series."

- Change of 7.4.2.3 (mandatory requirement in the main part of the standard IEC 61508-1):

  7.4.2.3 The hazards, hazardous events and hazardous situations of the EUC and the EUC control system shall be determined under all reasonably foreseeable circumstances (including fault conditions, ***reasonably foreseeable misuse and malevolent or unauthorised action****). This* ***shall include all relevant security, cyber security and human factors issues****, and shall give particular attention to abnormal or infrequent modes of operation of the ECU.*

  Detailed requirements and recommendations for implementation of appropriate security countermeasures and the underlying processes should be handled by cybersecurity standards like IEC 62443 or the ISO 27000 series, and co-engineering as expressed in IEC TR 63069 "Framework for functional safety and security", section 6, "Life cycle recommendations for co-engineering" This section was mainly driven by AIT guided by work in the ECSEL projects AMASS and AQUAS.

  Note: EUC = Equipment under Control

The co-engineering aspect are covered as recommendation in IEC TR 63069, as mentioned above. Figure 14 describes the process of iteration and interaction between the safety and security domain independently from how it is organized within an organization (separate or joint safety and security departments etc.):

(*Figure 14 is taken from IEC TR 63089, chapter 6, "Life Cycle recommendations for Co-engineering"*).

It should be noted that just now work has started on upgrading the TR 63069 to a TS (Technical specification), which is already ranked as an international standard containing mandatory requirements (normative) and not only recommendations (informal).

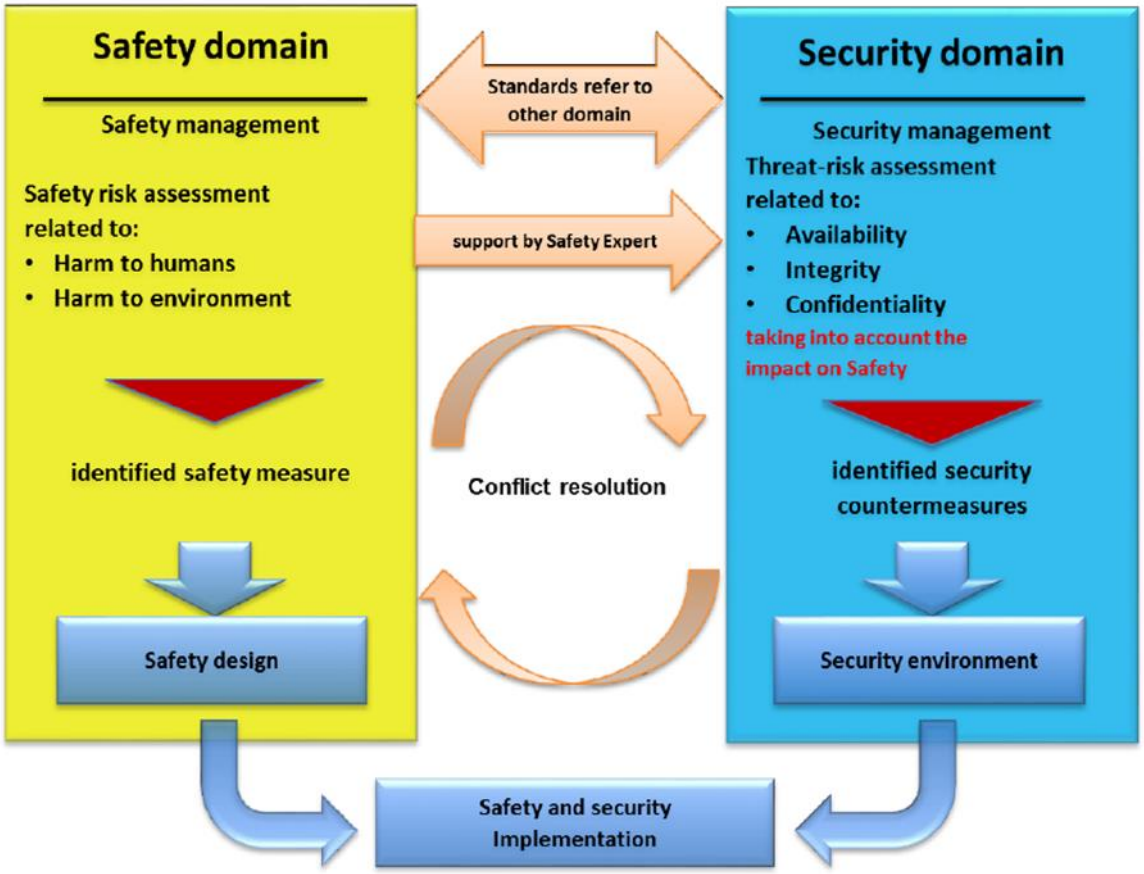

Figure 14: Safety and security co-engineering as described in IEC TR 63069

This approach is supported by description of ”Three guiding principles” (citation):

1) Guiding principle 1: protection of safety implementations

Security countermeasures should effectively prevent or guard against adverse impacts of threats to safety-related systems and their implemented safety functions. Evaluations of safety functions should be based on the assumption of effective (security) countermeasures.

EXAMPLE 1:

- Security countermeasures are expected to prevent unauthorized modification of safety relevant software, e.g. via remote access.
- Security related investigation of safety software/code or other processes related activities prevent against unintended implementation of malware in safety critical code.

2) Guiding principle 2: protection of security implementations

The safety measures should not have adverse impact on the effectiveness of security implementations.

NOTE: Human factors are taken into account from perspectives of both safety domain and security domain.

EXAMPLE 2

- Safety installations are not allowed to add features, e.g. remote access to systems, not being assessed by security.

- Safety functions might be more sensitive related to DoS (denial of service) attacks and therefore being a potential target to adversely affect the availability of a system.
- Guiding principle 3: compatibility of implementations

3) Security implementations and safety implementations should not have adverse contradictions.

EXAMPLE 3

- The communication speed of a system is affected due to security countermeasures and therefore adversely affecting the timing aspects of the safety function.
- Cryptographic methods used for security are not allowed to adversely affect the communication channel 610 protection measures used by safety.

Looking at risk mitigation for functional safety and security, there is no pre-defined priority.

Now IEC TC65 WG20 is working on further exploration of the framework standardization by having initiated work towards a TS, which is, as opposed to a TR, not only a recommendation, but also normative. An important proposal was submitted at the last Web Conference not to restrict only on the functional safety aspect but to extend to safety in general, which would simplify the discussion of better integrating the co-engineering aspects between different dependability domains.

4.2.2 Objective 10: Promoting awareness

To promote awareness and bring results of AQUAS into at least two international standards in the functional safety and security area with respect to safety, security and performance co-engineering.

As noted previously, awareness-raising initiatives are valid alternatives even in the absence of perfect synchronisation with revision cycles, and AQUAS partners are currently involved in various initiatives of these kinds.

Partner AIT presented the AQUAS approach of interaction points in a meeting aimed at the development of **ISO/SAE 21434** as a potential approach how to instantiate the communication channel and use them in applied processes. Communication channels from safety to security are currently already required in the FDIS version of ISO26262 Edition 2. The challenge is that the current definition of “communication channel” is based on exchanged information and there is no process specifying *how* such information could be exchanged. The interaction points [21][22] from AQUAS are a potential way to specify how a communication channel could be defined in a company-specific process. The same was done by AIT in maintenance meetings for **IEC MT61508-1/2** and **MT 61508-3**, and for **IEC 63069** (“Framework for functional safety and security”).

AQUAS partner AIT is involved in this area, where standards are being set up in **IEC TC 65 on Smart Manufacturing** and in **IEC TC 65 SC65A in updating IEC 61508 towards Ed. 3**. Preliminary work has started, and the official stability phase of Ed. 2, 2010, was extended to have more time to prepare a solid Ed. 3 considering new paradigms (particularly in software) and on transversal topics like co-engineering or consideration of cybersecurity in safety standards. The TC65 AhG1, later known as WG20, together with TR 63069, tried to fill the gap in “Framework for functional safety and security”. The co-engineering aspect (a subchapter) and the interaction between safety and security teams are described, but it was a compromise: contrary to the AQUAS approach, the interaction is not described in a fully symmetric manner. For preparation of the next edition of IEC TR 63069, AIT presented December 2018 at the last F2F-Meeting of IEC TC65 WG20 the concepts of AQUAS, AMASS and of an experience report of an industrial member of the mirror committee of IEC TC65 in Austria, ÖVE MR65, from petrochemical industry (ÖMV). It was well received by the working group members.

Also, AMT via its parent company ANSYS Inc. as a member of the standardisation committee is influencing the modernization of the **IEC 61508**. The AQUAS principle of performing model-based

analysis, development and verification was presented to the committee and is proposed to be an informative annex of the standard. Furthermore, AMT takes part in the discussion whether normative requirements related to cybersecurity should be defined in the standard or not.

To speed up the process for a modernized IEC 61508 as a system standard, WG 18 was created in SC65A for **IEC 63187**, "Functional safety - Framework for safety critical E/E/PE systems for defence industry applications", which plans to overcome the IEC 61508 weaknesses versus large systems, systems engineering concepts, and concepts like the interaction between safety & security (this is already marked as a separate chapter). The cybersecurity task group of **IEC MT 61508** (Joint between MT 61508-1/2 and MT 61508-3) could not find consensus on three proposals, one trying to reduce details on security management and interaction points within the functional safety standard, the other two strengthen this link in a mandatory manner. The three proposals were tabled and had to be commented on by the national committees of members until October 31st, 2019, and be discussed until the meeting in Vienna December 16-20, 2019, in Vienna. AIT has submitted statements and comments on the three proposals with clear focus on strengthening mandatory requirements on co-engineering aspects and interaction between functional safety and cybersecurity, supporting the most extensive proposal in this direction.

In the area of safety resp. performance AQUAS partner AMT contributes via its parent company ANSYS Inc. to **ARP4761A**, the safety standard in the avionics domain and to **ISO 21448** – Safety of the intended function (SOTIF). In both standardization activities ANSYS Inc. is a member of the standardization committee. The main contribution to ARP4761A is the definition of a concept for model-based safety analysis (MBSA). Regarding ISO 21448 (see 2.3.1) ANSYS Inc. takes effort in defining the interface between the SOTIF-specific analysis methods and the validation and verification (V&V) by means of simulation which. An iterative and oriented V&V strategy was proposed to Annex A.1 of the standard. This interface forms an Interaction Point in the sense of the AQUAS project.

Based upon co-engineering in the AQUAS project, feedback from AQUAS consortium (two meeting concerning gap analysis), response from a member of standardization committee Kevin Staggs and experience from the real implementation of security by TrustPort practitioners of this domain, the gap analysis of **ANSI/ISA-62443-3-3** security standard was published in [66]. This gap analysis revealed the missing part of ANSI/ISA-62443-3-3 security standard. Based on this analysis, the possible recommendations for extending 62443-3-3 are proposed. These recommendations are discussed and proposed to related standard committee.

Thanks to co-engineering AQUAS approach using an extensive Secure Software Development Life Cycle catalogue containing security requirements together with the advanced modelling framework TTool based on UML/SysML-Sec for performance analysis, the related standard committee were proposed new verifications methods and extensions of security requirements with performance trade-offs.

Further collaboration activities were focused on **IEC 61508**, where effort has been shared in the following way:

- Coordination (phone conferences of the AQUAS IEC 61508 team, gap anlysis, standards evolution presentation of IEC 61508)
- Presenting dependability co-engineering to related committee (IEC TR 63069, Framework for functional safety and security, human factors task group of IEC MT 61508)
- Review of differences with recommendations for evolution (AQUAS IEC 61508 gap analysis, IEC 63187 and IEC MT 61508 JTG 19 analysis, planned new TR on deficiencies of current IEC 61508 because a structural change cannot be made in the time frame of Ed. 3 (CD planned Spring 2021))
- Proofs/examples/references of AQUAS work.

- Overall findings to be presented to the consortium (AQUAS gap analysis IEC 62443 and 61508, (Done in IEC TR 63069, IEC 61508 plus IEC 63187 in IEC 61508 team phone conferences)
- Change request of findings to the related committee (or otherwise inform). (Done during the last years in IEC committee meetings of all of the standards mentioned above, mainly in discussions/contributions)

A thorough gap analysis on IEC 61508 was provided by ITI, which complements the general gap analysis done in the work on **IEC 63187**, "Functional safety - Framework for safety critical E/E/PE systems for defence industry applications" (WG 18 of IEC SC65A, same committee as responsible for IEC 61508), where AIT is active. These activities were discussed in the IEC 61508 team phone conferences lead by AIT. The valuable outcomes will be used as further input to IEC 63187 and a planned TR (Technical report), which was proposed in a Joint Task Group of IEC MT 61508-1/2 and MT 61508-3. The key elements of these considerations are:

- Historical reason: concept strongly influenced by Process Industry
- Main weakness: Not integrated in general system engineering

The system engineering aspect includes:

- From a system engineering point of view
  - Safety is only one aspect
  - Safety issues are tackled by dedicated processes and activities
  - Not described in IEC 61508 – only outcomes are requirements
- From a safety point of view
  - Surrounding processes and activities are necessary
  - No reference to ad-hoc standards is given in IEC 61508
- This is a problem for sectors
  - Understanding IEC 61508 as a handbook
  - Lacking maturity on system engineering
- Consequence
  - Most implementations in medical devices, process and manufacturing industries lack the references to ad-hoc engineering frameworks, leading in most cases to failure of the compliance
  - Daughter standards of concerned sectors reflect this weakness

This AQUAS gap analysis is planned to bring further forward to this Task Group to improve the concept with respect to co-engineering of multi-concern assurance issues.

In the **automotive standardization landscape**, the groups covering functional safety and SotiF (Safety of the intended Functionality) are rather cooperative with the Cybersecurity engineering group – ISO 26262 had already (proposed by AIT) a link to security within *one mandatory requirement (similar as now done in IEC 61508) and additionally an Annex containing a concept of interaction points*, and several cybersecurity related standards (ISO/SAE 21434, ISO 24089 (Software update/Over the Air Update) and NP 5112 (Audit guidelines for cybersecurity engineering, security management throughout the supply chain, led by AIT as chair). This is another success for raising successfully awareness of the need of co-engineering like interaction for safety and cybersecurity actors.

4.2.3 Objective 11: Influencing framework-oriented standardisation groups

To influence actively in two international standardization groups focused on frameworks for the coordination of safety, security, and reliability of automation.

In **Smart Manufacturing**, an Ad-hoc group **AhG3** was started in IEC TC65, which delivered a report and starts as WG 23. Together with ISO TC 184 (Automation systems and Integration), JWG 21 was founded on Smart Manufacturing – Reference models. A Cybersecurity Task Force was founded for AhG3 to look at the safety – cybersecurity issues in context of smart manufacturing. However, the safety group is separate (and not so active), which is a little bit unfortunate because there is almost no interaction between these groups. Here again, AQUAS Partner AIT is trying to raise awareness and get consensus on the co-engineering and interaction point concepts – but it is a hard fight to achieve consensus in this direction.

CITY presented gap analysis at the Human Factors and Ergonomics Society, Europe, Annual Conference 2019 on November 2019 in Nantes, France. The gap analysis addresses the following standards:

- **EN 62366-1:2015** (Application of Usability Engineering to Medical Devices),
- **EN 62366-2:2016** (Guidance on the Application of Usability Engineering to Medical Devices),
- **EN 60601-1-6:2010** (General Requirements for Basic Safety and Essential Performance - Collateral Standard: Usability
- **EN 60601-1-8** (General Requirements for Basic Safety and Essential Performance - Collateral Standard: General requirements, tests and guidance for alarm systems in medical electrical equipment and medical electrical systems.
- **EN 60601-1-10** (General Requirements for Basic Safety and Essential Performance - Collateral Standard: Requirements for the development of physiologic closed-loop controllers).

The gap analysis presented in Nantes was later extended, especially with examples from current research and incident reports, and to include a comparison, not just with similar standards from other domains (aviation), but also medical human factors standards in other countries (HE 75 standard). The extended work was presented at the Health Informatics Conference 2020 in Malta and published in the conference proceedings.

Finally, this work was also presented, for internal reporting, to members of the AQUAS consortium in May 2020, and to staff at City University, London in February 2020.

ARCADIA represents a standardised methodology for System Engineering: Final status and partner team activities achieved during the project:

- The gap analysis (recommended evolutions) was accomplished - the study is available in the Appendix. A key outcome is that the AQUAS approach has initially more evolution to make than ARCADIA, but the two approaches appear to be very complimentary. A general recommendation for ARCADIA was to have wider coverage of the product life cycle.
- Action to present AQUAS to the ARCADIA working group: Emails exchanged about the activity with working group members. We also included a member of the working group onto the AQUAS advisory board who participated at workshops (Thales-UK).
- There were two aspects unachieved by the team. We found the modelling tool Papyrus can align well with Arcadia, but the existing capabilities of Papyrus for DCE is not yet clear. The other aspect that would have been desirable to consider relates to linking the findings of the gap analysis with challenges of the demonstrators.
- Presenting findings to the AQUAS consortium: We did this in a videoconference where all partners were invited. It was well received without changes recommended. We would have had higher coverage at a physical meeting – but this option was unavailable at the time.
- Publish findings to the Arcadia committee. The gap analysis has been circulated to members of the working group and to be included in a planned meeting between the Thales Divisions.

Of course the findings of the study is that DCE has more changes to make than ARCADIA in relation to integration.

Finally, within the framework-oriented standards on the **Internet of Things** (see also Section 2.3) and **AI (Artificial intelligence)**, Austrian partners are active mainly on national level, where influencing development towards common safety and security issues remains a challenge at this stage, but is now addressed within the "Trustworthiness" concept.

4.2.4 Objective 12: Promoting awareness in other standardisation groups

To promote awareness and bring results of AQUAS into at least two other engineering international standards, such as OMG, or FMI.

MARTE 2.0

A major action is currently underway for the Revision 2.0 of MARTE OMG standard, that has involved the preparation of Request For Information (RFI) for MARTE 2.0, envisioned as an extended version with capacities to manage complementary DSLs, plus various technologies to serve IoT domain applications and other kinds of specialised analysis for non-functional properties in conjunction with software and hardware design.

AQUAS partners have been involved in this initiative since the beginning, to create opportunities to bring AQUAS results in the MARTE standard.

Over the first year of life of AQUAS, we have progressed mostly in the collection and submission to OMG of the interests of partners and the definition of the AQUAS high-level requirements for MARTE 2.0. This activity has been coordinated internally by INT and TRT, with interest expressed by CEA, INT, MAGILLEM, MTTP, UNIVAQ, ITI, TRT and CITY and also coordinated externally with the MegaM@Rt2 and the AMASS ECSEL projects.

As a result, AQUAS partners CEA, TRT and Intecs have submitted their answer to the RFI for MARTE 2.0 due by 15 August 2019. The principal requirements gathered to date concern the expansion of its modelling and annotation capabilities for current evolution of real-time embedded systems (e.g. CPS, IoT, and Industry 4.0) and their necessary quality attributes (with emphasis on dependability, safety, and security attributes) (see synthesis in section 3.2.10. Partner TRT has submitted the AQUAS requests during a MARTE OMG meeting held in September 2019.

RFI submissions will help OMG to prepare the initial draft for the MARTE 2.0 Request for Proposal (RFP). As OMG members CEA and TRT AQUAS partner's plans also to participate in an initial submission team that responds to the RFP. If possible, they may even contribute to the revised submission in the course of the project.

SysML

AMT as part of the ANSYS Inc. is member of the SysML2 Submission Team (SST) which harmonizes all work before the initial submission for SysML2. The special objective of this work is to specify modelling concepts for safety and cybersecurity by means of plugins or language extensions.

VEL by OASIS

TEC, UNIVAQ and MTTP have collaborated to identify the needs for VEL to support trade-off analysis among quality attributes as reported in section 3.2.8 and submitted them to the VEL community chairs to get feedback.

IP-XACT

Magillem is co-chair of IP-XACT, and is contributing to the definition of the new release of IP-XACT, to be released in 2020. This contribution may include needs coming from AQUAS, to support the Interaction Point concepts.

A collaboration among Magillem and SAG is defined to motivate the need for extension, propose an application use case, and provide the corresponding IP-XACT files using the vendor extensions mechanism. A technical document will detail the proposed extension description into the schema.

MISRA C

Within AQUAS, AbsInt and SYSGO are working together to adapt AbsInt's Astrée/RuleChecker to the analysis of SYSGO's PikeOS operating system. In doing so, they came up with a system of coding rules for operating-system code with the same spirit as the MISRA rules, but differing from them in certain details to the effect that there are fewer undesired violations and the certification work is reduced. Dr. Daniel Kästner from AbsInt is a member of the MISRA C Working Group and leverages these findings for his contribution to the Working Group. There are five group meetings per year, the most recent one took place on October 9 and 10, 2019. During the COVID-19 pandemic, the working group meets online every 14 days. AbsInt participated at the MISRA group meeting in October 2019 and in all online meetings currently organized in order to, among other things, bring attention to lessons learnt within AQUAS.

FMI

The return of experience of integrating executables controllers as FMI raised questions regarding the integration methods, split or the logical interfacing (what is on the controller side and what is on the model side) and also limitations of the FMI interfaces to support new cases never addressed before. The convergence with some SystemC definition is also a question to be raised to allow future virtual platform integration via FMI. Siemens will address these points in its discussions with the FMI steering committee.

5 Other Activities

This section describes a set of internal activities that are intended to keep the standards evolution activity focused and on track throughout the project.

5.1 Identification of key participants and skill sets

As part of the overall initiative to track progress key participants from the consortium (in terms of both person/months available for commitment to the activity and skills / contacts available for this activity) were identified, so that maximum involvement over the consortium can be achieved, and opportunities for exploiting consortium-internal resources are not missed. This is being managed in the spreadsheets made available consortium-wide for this purpose.

5.2 Tracking progress toward objectives

In order to channel upcoming activities in the remainder of the project into the most effective paths, both the expert advisory board and the project reviewers have recommended to put into place mechanisms for tracking progress toward the achievement of the objectives of the project activities. This is currently being implemented for the Standards Evolution Goal in the harmonized spreadsheets over the three project goals.

In particular, a set of preliminary general challenges in the area of the Standards Evolution goal was identified, as illustrated in Table 5.

Challenge	Progress Indicator
How to provide visibility of challenges and progress, addressing priorities and decisions (supported in AQUAS or later).	Number of presentations either in AQUAS related meetings (e.g. EAB) or public conferences
Industry may have reservations to adopt an approach which is not reflected in current standards.	Number of explicit contacts established with companies on the question of standards-based co-engineering
There are domains in which integrated approaches to safety and security are not fostered by the governing standards –or even implicitly discouraged.	Number of papers or public reports (including AQUAS deliverables) arguing integrated standards approaches

Table 5: Preliminary progress indicators against general SE challenges

In the second half of Y2, 5 specific progress indicators steps against the SE objectives were finally identified, as in the following:

1. Identification of Standard(s) to be addressed, depending on:
    a. Window opportunity- considering the revision cycles of the specific standard, or the standardization committee being open to receive recommended best practices;
    b. Contact opportunity - involvement of partners standardization committee, or availability of contacts with standardization committee participants or convenors;
    c. Gap analyses –identification of AQUAS evolution needs;
2. Definition of a collaborative activity focussed on the specific standard (possibly with participation of a team of partners)
3. Establish contact with standardization group(s)
4. Standard(s) Review with Gap Analysis

5. Submission (in terms of any of the following):
    a. Request for change(s)
    b. Presentation at the standards committee of the need for evolution
    c. Dissemination of Gap Analysis
    d. Submission of Guidelines
    e. Recommended best practices

Tracking of the progress was implemented for the Standards Evolution Goals in the project repository through harmonized spreadsheets detailing opportunities and steps performed over the 4 objectives.

A synthesis of the current status and quantification of the achievements at M38 is reported in Annex A.

5.3 Harmonization of terminology

Another important activity launched over all three AQUAS goals was the **harmonization of terminology**. This was also the result of a recommendation from the external advisory board: ensure that any proposals resulting from AQUAS work are consistent with the directions being taken by the standardisation groups, in order to avoid the ugly surprises that result from extreme mismatches both at the conceptual and at the terminological levels.

These efforts were conducted in this direction in Y2 of the project, reacting to first external advisory board recommendations. Table 6 presents an extract from the (much larger) table currently being maintained, in an attempt to arrive at a proposed harmonisation of terminology.

As can be seen in the table, there are already problematic areas. For example, “risk” is defined and understood in many different ways in different standards, and it will be a challenge for AQUAS to arrive at an operational understanding of risk that is sufficiently robust to support its methodology and at the same time satisfy the interpretations of the target standards that AQUAS will later approach. Nevertheless, the AQUAS consortium recognizes the importance of the recommendation to analyse these terminological issues as an integral aspect of a viable initiative to influence standards.

Another problematic area is the terminology involving the **performance** parameter. Hardly any standards to date even define performance as part of any formal glossary. Only with the advent of the most recent standards governing autonomous applications (e.g. aerial drones and self-driving road vehicles) are we beginning to see the performance parameter inserted into controlled vocabularies – and even then, there is only a top-level, relatively generic definition. Clearly, this parameter in particular is only beginning to be treated systematically in standards, and the AQUAS project could be instrumental in formalizing its place in co-engineering.

Term	Definition	Type	Source
Safety	State where an acceptable level of risk is not exceeded. This may apply to the system or its environment (particularly to people).	Safety	ECSS / CRR
Risk	The level of impact on organizational operations (including mission, functions, image, or reputation), organizational assets, or individuals resulting from the operation of an information	Transverse	FIPS 200

	system given the potential impact of a threat and the likelihood of that threat occurring.		
Safety Integrity Level	Discrete level, corresponding to a range of safety integrity values	Safety	IEC 61508
Security level	Level corresponding to the required effectiveness of countermeasures and inherent security properties of devices and systems for a zone or conduit based on assessment of risk for the zone or conduit	Security	IEC 62443
Performance limitation	Insufficiencies of the function itself	Performance	SOTIF
Trade-off	Decision-making actions that select from various requirements and alternative solutions on the basis of net benefit to the stakeholders	Transverse	ISO/IEC 15288:2015

Table 6: Extract from terminology harmonisation table

6 Conclusions

The submission of version 1.0 of this this deliverable occurred at the halfway of Y3 in the AQUAS project. The current version 2.0 update extends the reporting on the final progress towards the achievement of the standards evolution objectives at M38.

As the previous discussions in this document have illustrated, contacts and submission of requests for evolution continued after M30 and new ones are being established until M38, as well as new initiatives were conducted. Section 4.2 of this document provides an update and progress of these specific initiatives, indicates which partners are involved, and finally describes how the project has produced a large number of results and elicited actionable requirements for standards evolution.

As noted in the introduction to section 4, the fact that AQUAS partners have established collaboration activities, analysed gaps and have already submitted or are well positioned to promote standard evolution, does not guarantee that all this will happen within the end of the project, or in the short time, depending on the revision cycles of the specific standards. However, the strategies outlined were employed regardless of revision cycles and ensured that AQUAS partners were able to produce a set of change requests, influence standards and produce awareness, in the project time frame, with effects that will definitely span beyond the lifetime of the project.

7 References

[1] "SWIM-TI Yellow Profile Technical Specification 3.1", 14.01.04.D43-004, 00.01.00, December 2015.

[2] "SWIM-TI Blue Profile Technical Specification 3.1", 14.01.04.D43-005, 00.01.00, December 2015.

[3] "SWIM-TI Purple Profile Technical Specification 3.1", 14.01.04.D43-006, 00.01.00, December 2015.

[4] D2.2.1 – Demonstrator Architecture – Air Traffic Management, AQUAS project, http://aquas-project.eu, 31.03.2018.

[5] D2.2.2 – Demonstrator Architecture – Medical, AQUAS project, http://aquas-project.eu, 31.03.2018.

[6] D2.2.3 – Demonstrator Architecture – Railways, AQUAS project, http://aquas-project.eu, 31.03.2018.

[7] D2.2.4 – Demonstrator Architecture – Industrial Drive, AQUAS project, http://aquas-project.eu, 31.03.2018.

[8] D2.2.5 – Demonstrator Architecture – Space Multicore, AQUAS project, http://aquas-project.eu, 31.03.2018.

[9] https://ec.europa.eu/growth/single-market/european-standards/harmonised-standards/medical-devices_en

[10] IEC 62443, Industrial communication networks – Network and system security, 2010.

[11] IEC 61508:2010 (ed. 2), Functional safety of electrical/electronic/programmable electronic safety-related systems, 2010.

[12] IEC 61800, Adjustable speed electrical power drive systems, 2015.

[13] ISO26262:2018 (ed. 2) – Road Vehicles – Functional Safety, 2018.

[14] ISO PAS 21448:2019 – Safety of the Intended Functionality, 2019.

[15] D2.3 – Safety and Security Balanced Mechanisms, SESAMO project, http://sesamo-project.eu, 10 November 2014.

[16] D2.1 – Security and Safety Modelling, SESAMO project, http://sesamo-project.eu, 29 May 2013.

[17] D3.2 – Combined Safety, Security and Performance Analysis and Assessment Techniques – preliminary, May 2019.

[18] Medini Analyze, ANSYS medini Technologies AG, http://www.medini.eu/.

[19] CHESS, INTECS, http://www.intecs.it/eng/rd_dettagli.asp?ID_RD=5.

[20] D2.1.4 – Domain Environment Industrial Drives, AQUAS project, http://aquas-project.eu, 31.10.2017.

[21] Various documents and discussions on interaction points, https://svn.trt.thalesgroup.com/repos/aquas/Work_Packages/WP3%20-%20Methodology/WP3-methodDiscussions

[22] How to specify interaction points, https://svn.trt.thalesgroup.com/repos/aquas/Work_Packages/WP3%20-%20Methodology/WP3-methodDiscussions/interactionPointScheduling_v02.pdf

[23] MALINA, L.; HAJNÝ, J.; FUJDIAK, R.; HOŠEK, J. On Perspective of Security and Privacy- Preserving Solutions in the Internet of Things. Computer Networks, 2016, vol. 102, no. 2016, p. 83-95. ISSN: 1389-1286.

[24] OMG, The OMG Hitchhiker's Guide - A Handbook for the OMG Technology Adoption Process, OMG Document: omg/2008-09-02, version 7.8, 2008.

[25] https://www.incose.org/docs/default-source/aboutse/se-vision-2025.pdf

[26] INCOSE. 2015. Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, 4th Edition

[27] OASIS VEL Technical Committee https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=vel

[28] Michael Schulze, Robert Hellebrand "Variability Exchange Language- A Generic Exchange Format for Variability Data" In Software Engineering (Workshops), Dresden, Germany, 2015, pages 37-46

[29] Damir Bilic, Etienne Brosse, Andrey Sadovykh, Dragos Truscan, Hugo, Bruneliere, Uwe Ryssel "An Integrated Model-based Tool Chain for Managing Variability in Complex System Design" Models and Evolution Workshop (ME 2019) at MODELS 2019

[30] Richard Pohl, Mischa Höchsmann, Philipp Wohlgemuth, Christian Tischer "Variant management solution for large scale software product lines", ICSE-SEIP 2018

[31] L. Pomante, V. Muttillo, B. Krena, T. Vojnar, F. Veljkovic, P. Magnin, M. Matschnig, B. Fischer, J. Martinez, T. Gruber "The AQUAS ECSEL Project Aggregated Quality Assurance for Systems: Co-Engineering Inside and Across the Product Life Cycle", Journal on Microprocessors and Microsystems Volume 69, MICPRO 2019

[32] R. Fujdiak and P. Mlynek and P. Blazek and M. Barabas and P. Mrnustik "Seeking the Relation Between Performance and Security in Modern Systems: Metrics and Measures" 41st International Conference on Telecommunications and Signal Processing (TSP) 2018

[33] L. Apvrille, Letitia LI, "Harmonizing Safety, Security and Performance Requirements in Embedded Systems". Proceedings of the DATE 2019 conference, Florence, Italy.

[34] J. Teich. "Hardware/software codesign: The past, the present, and predicting the future". Proceedings of the IEEE, 100(Special Centennial Issue):1411–1430, May 2012

[35] A. Osyczka. "Multicriteria optimization for engineering design". A. Osyczka (Ed.), Design Optimization, Academic Press. 1985, pp. 193-227

[36] Mukelabai Mukelabai, Damir Nešić, Salome Maro, Thorsten Berger, and Jan-Philipp Steghöfer "Tackling combinatorial explosion: a study of industrial needs and practices for analyzing highly configurable systems" In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE 2018), 2018

[37] Lina Ochoa, Oscar González-Rojas, Alves Pereira Juliana, Harold Castro, Gunter Saake "A systematic literature review on the semi-automatic configuration of extended product lines" Journal of Systems and Software, 2018

[38] Alberto Allara, Massimo Bombana, William Fornaciari, and Fabio Salice. "A case study in design space exploration: The tosca environment applied to a telecommunication link controller". IEEE Design & Test of Computers, 17:60–72, 2000

[39] V. Reyes, T. Bautista, G. Marrero, P. P. Carballo, and W. Kruijtzer. Casse: a system-level modeling and design-space exploration tool for multiprocessor systems-on-chip. In Euromicro Symposium on Digital System Design, 2004. DSD 2004., pages 476–483, Aug 2004.

[40] Hector Posadas, Pablo Peil, Alejandro Nicols, and Eugenio Villar.” Automatic synthesis of communication and concurrency for exploring component-based system implementations considering UML channel semantics”. Journal of Systems Architecture, 61(8):341 – 360, 2015.

[41] V. Zaccaria, G. Palermo, F. Castro, C. Silvano, and G. Mariani. “Multicube explorer: An open source framework for design space exploration of chip multiprocessors”. In 23th International Conference on Architecture of Computing Systems 2010, pages 1–7

[42] R. Görgen et al., "CONTREX: Design of Embedded Mixed-Criticality CONTRol Systems under Consideration of EXtra-Functional Properties," 2016 Euromicro Conference on Digital System Design (DSD), Limassol, 2016, pp. 286-293.

[43] Leire Etxeberria Goiuria Sagardui and Lorea Belategi. 2008. “Quality aware software product line engineering,” Journal of the Brazilian Computer Society. March 2008, Volume 14, Issue 1, pp 57–69.

[44] David Benavides, Sergio Segura, and Antonio Ruiz-Cortés. 2010. Automated analysis of feature models 20 years later: a literature review. Information Systems 35, 6 (2010), 615–708

[45] Norbert Siegmund, Marko Rosenmüller, Martin Kuhlemann, Christian Kästner, Sven Apel, Gunter Saake "SPL Conqueror: Toward optimization of non-functional properties in software product lines"

[46] Zhang, Guoheng, Huilin Ye and Yuqing Lin. “Modelling Quality Attributes in Feature Models in Software Product Line Engineering.” ICSOFT (2011).

[47] Rafael Olaechea, Steven Stewart, Krzysztof Czarnecki, and Derek Rayside. "Modelling and multi-objective optimization of quality attributes in variability-rich software". In Proceedings of the Fourth International Workshop on Nonfunctional System Properties in Domain Specific Modeling Languages (NFPinDSML '12) 2012

[48] Alexandr Murashkin, Michał Antkiewicz, Derek Rayside, and Krzysztof Czarnecki. 2013. Visualization and exploration of optimal variants in product line engineering. In Proceedings of the 17th International Software Product Line Conference (SPLC '13) 2013

[49] Gustavo G. Pascual, Roberto E. Lopez-Herrejon, Mónica Pinto, Lidia Fuentes, Alexander Egyed, "Applying multiobjective evolutionary algorithms to dynamic software product lines for reconfiguring mobile applications" Journal of Systems and Software, Volume 103, 2015

[50] Chistopher Henard, Mike Papadakis, Mark Harman, and Yves Le Traon “Combining Multi-Objective Search and Constraint Solving for Configuring Large Scale Software Product Lines” ICSE 2015

[51] Juliana Alves Pereira "Search-Based Product Configuration in Software Product Lines" Master thesis, 2014

[52] TTool, http://ttool.telecom-paristech.fr

[53] https://www.sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

[54] van der Peijl, J., Klein, J., Grass, C., & Freudenthal, A. (2012). Design for risk control: the role of usability engineering in the management of use-related risks. Journal of biomedical informatics, 45(4), 795-812.

[55] Alberdi, E., Strigini, L., Povyakalo, A. A. & Ayton, P. (2009). Why Are People's Decisions Sometimes Worse with Computer Support? Computer Safety, Reliability, and Security, Proceedings, 5775, 18 - 31.

[56] A Guide to the Project Management Body of Knowledge – PMBOK Guide Fifth Edition, 2013 (about 580 pages)

[57] ISO 21500 Guide on Project Management (was modelled around PMBOK although there are some differences)

[58] The Human Factors Analysis and Classification System –HFACS – Scott A. Shappell, FAA Civil Aeromedical Institute Oklahoma City, OK 73125 Douglas A. Wiegmann; University of Illinois at Urbana-Champaign Institute of Aviation Savoy, IL 61874; February 2000 Final Report. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.

[59] ESA Guide for Independent Software Verification and Validation, Edition 2, 2008

[60] N. Silva and R. Lopes, "10 Years of ISVV: What's Next?," 2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops, Dallas, TX, 2012, pp. 361-366.

[61] Goodloe, A.: Challenges in high-assurance runtime verification. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 446–460. Springer, Heidelberg (2016)

[62] Kosmatov, N., Marché, C., Moy, Y., Signoles, J.: Static versus dynamic verification in Why3, Frama-C and SPARK. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I, LNCS, vol. 9952, pp. 461–478. Springer, Heidelberg (2014)

[63] Cazorla, Francisco J. ; Kosmidis, Leonidas ; Mezzetti, Enrico; Hernandez, Carles; Abella, Jaume ; Vardanega, Tullio: Probabilistic Worst-Case Timing Analysis: Taxonomy and Comprehensive Survey. In: ACM Comput. Surv. 52 (2019), Februar, Nr. 1, 14:1–14:35. http://dx.doi.org/10.1145/3301283. – DOI 10.1145/3301283. – ISSN 0360–0300

[64] Voirin, Jean-Luc. Model-based System Engineering with the Arcadia Method. Oxford: Elsevier, 2018.

[65] Pasquinelli, Mauro, Luis Molina-Tanco, Arcadio Reyes-Lecuona, and Michele Cencetti. "Extending the System Model." In Dynamics of Long-Life Assets, by Stefan N. Grösser, Arcadio Reyes-Lecuona and Göran Granholm, 169-189. Cham: Springer, 2017.

[66] Petr Mlynek, Radek Fujdiak, Pavel Mrnustik, Bohuslav Krena, Ludovic Apvrille. Co-Engineering Gap Analysis of ANSI/ISA-62443-3-3. 2020 (in print in DOI 10.11601, ISSN1805-5443 - http://www.ijates.org/index.php/ijates).

8 Glossary

AIOTI	Alliance for Internet of Things Innovation
ATM	Air Traffic Management
AQUAS	Aggregated Quality Assurance in Systems
CE	Co-Engineering
CENELEC	European Committee for Electrotechnical Standardization
CO	Confidential
COTS	Commercial Off The Shelf
CPS	Cyber Physical System
DKE	Deutsche Kommission Elektrotechnik Elektronik im DIN und VDE
DSL	Domain Specific Language
ECSS	European Cooperation for Space Standardization
EN	European Norm
ESA	European Space Agency
ETSI	European Telecommunications Standards Institute
EUC	Equipment under control
FPGA	Field Programmable Gate Array
HTTP	Hyper Text Transfer Protocol
IACS	Industrial automation and control system
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
INCOSE	International Council on Systems Engineering
IOT	Internet of Things
ISO	International Standards Organization
JWG	Joint Working Group
MARTE	Modelling and Analysis of Real-time and Embedded Systems
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
OSLC	Open Services for Lifecycle Collaboration
OTA	Over the Air Updates
PAS	Publicly Available Specification
PLC	Product life-cycle
PLM	Product Lifecycle Management
RDF	Resource Description Framework

RFI	Request For Information
RFP	Request for Proposals
RRM	Risk reduction measures
SAE	Society of Automotive Engineers
SDO	Standards Developing Organisation
SEBoK	Systems Engineering Body of Knowledge
SESAR	Single European Sky ATM Research
SIL	Safety integrity level
SOTIF	Safety of The Intended Functionality
SPARQL	SPARQL Protocol and RDF Query Language
SRS	Safety-related system
SSP	Safety, Security and Performance
S/S/P	Safety / Security / Performance
SWIM	System Wide Information Management
SysML	Systems Modeling Language
TC	Technical Committee
UAV	Unmanned Airborne Vehicle
UML	Unified Modeling Language
URI	Universal Resource Identifier
VEL	Variability Exchange Language
WG	Working Group
WP	Work Package

Annex A – SE Objectives - Key Progress Indicators

A synthesis of the major results contributing to the achievement for the 4 Objectives for the Standards Evolution Goals and their quantification according to the KPIs described in section 5.2 at M38 is reported in the Tables 5-8.

Table 7: Objective 9 Progress

Objective 9	Standard	Definition of activity	Contact	Gap Analysis	Submission	Measure
UC1 ATM	ARINC653	y	y	y	contacts	90%
UC2 Medicine	EN-60601-1-10	y	y	y	Y (conference, contacts)	90%
	ISO/IEEE11073-00103	y	Y	y		80%
UC3 Railway	*CENELEC TC9X, CLC/prTS 50701*	*y*	*y*	*y*	*y*	*100%*
UC4 Industrial Drive	*IEC 61508, IEC 62443, IEC TR 63069*	*y*	*y*	*y*	*y*	*100%*
	ANSI/ISA-62443-3-3 (99.03.03)-2013	y	y	y	y	100%
UC5 Space Multicore	ECSS E40	y	y	y	Y	90%
Overall Measure (1 for each UC)						**95%**

Table 8: Objective 10 Progress

Objective 10	Standard	Definition of activity	Contact	Gap Analysis	Submission	Measure
	ISO/SAE 21434	*y*	*y*	*y*	*y*	*100%*
	CENELEC TC9X	*y*	*y*	*y*	*Y*	*100%*
	IEC MT 61508	*y*	*Y*	*y*	*y*	*100%*
	ANSI/ISA-62443-3-3	y	y	y	*Y (conference)*	100%
	ISO 21448 SOTIF	y	y	y	y	80%
	ISO/SAE 21434	y	y	y	y	*100%*

Overall Measure (2 submissions)	**100%**

Table 9: Objective 11 Progress

Objective 11	Standard	Definition of activity	Contact	Gap Analysis	Submission	Measure
	IEC TC 65 on Smart Manuf.	*y*	*y*	*y*	*y*	*100%*
	Human Factor STDs	*y*	*y*	*y*	*Y (conference)*	*100%*
	ARCADIA	y	y	y	y	100%
Overall Measure (2 submissions)						**100%**

Table 10: Objective 12 Progress

Objective 12	Standard	Definition of activity	Contact	Gap Analysis	Submission	Measure
	MARTE 2.0	*y*	*y*	*y*	*y*	*100%*
	VEL by OASIS	*y*	*y*	*y*	*Y*	*100%*
	IP-XACT	y	Y	In progress		80%
	MISRA C	y	y	y	y	90%
	FMI	y	y	y		60%
	SysML	*y*	*y*			*40%*
Overall Measure (2 submissions)						**100%**