AQUAS SAFETY SECURITY PERFORMANCE

Deliverable 1.2

Report on the future challenges to be overcome for dependability co-engineering

ECSEL Joint Undertaking

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:	38 months
Deliverable No.:	D1.2
Document name:	Report on the future challenges to be overcome for co-engineering
Work Package	WP1
Associated Task	Task 1.2
Nature [1]	R
Dissemination Level [2]	PU
Version:	1.0
Actual Submission Date:	26-06-2020 (update)
Contractual Submission Date	31-03-2020 (Extension granted by ECSEL JU to 10-06-2020)
Editor: Institution: E-mail:	T.Gruber AIT Austrian Institute of Technology Thomas.gruber@ait.ac.at

[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)

Executive Summary

We provide here a guideline with recommendations to future projects for impacting advancements related to Dependability Co-Engineering (DCE). AQUAS through the ECSEL-JU funding has provided a very useful opportunity to explore the challenges related to the uptake of optimisation/automation to treat the interdependencies (coupling) for safety, security and performance (dependability) properties.

DCE potentially influences all the technology in a system and is the most influenced by processes and policies at organisation and society level. This intersection means DCE faces significant challenges from both sides, but also the potential to provide high gains including much more uptake of other technologies.

AQUAS represents the first large consortium to investigate a coordinated engineering approach supporting evolution towards applying DCE and in relation to the product lifecycle. It is a progressive collaboration framework for both short and longer-term needs that advances both the tools and the product process (for DCE). AQUAS has considered both technical and non-technical needs including enhancement of standards to be more adapted for cross mediation.

The project has been supported by an extensive external advisory board of 23 members. They have been involved in six workshops validating, extending and prioritising challenges. They have also been involved in proof reading reports (including this one).

Two of the key recommendations from discussions with the consortium and the advisory board are related to the sheer scale of scope – so there is a *need to establish a DCE domain* and also *a centralised technical coordination body*, necessary for prioritising actions so as to remove the main the bottlenecks to treat that will enable industrial uptake.

Whilst the project had a focus on methodology and tooling investigations, there has also been significant consideration of coordinated collaboration and conditions that support advancement of this integrative-type technology, particularly bringing the work into mainstream practice. Some of the observations here include:

- It is necessary to align the common foundations for automating DCE and to provide visibility of progress and advancement in automating DCE based on progress indicators.
- This requires centralised technical coordination providing direction on key challenges, proposal inputs, sustainable advancements through transfer of group efforts and lessons learned between related projects.
- Mechanisms and a culture to support an Integrative R&D Approach across the integration levels of a CPS (CPS system properties, CPS key functions supporting system properties, and higher-level CPS key functions like Smart Sensing or Processing) shall be promoted.
- Funding bodies should be guided to provide substantial support for these integrative approaches.
- Engagement with policy makers and regulation to reduce barriers on long-term type technologies.
- Publicly available material should be provided supporting companies in deploying DCE processes and technologies

There was significant work to advance the DCE associated technologies. Related to the AQUAS methodology:

- In AQUAS, measuring the actual cost reduction was limited to the development stages in the lifecycle; For this, a cost model was developed, which needs validation in industry. Measuring the savings in operation by avoiding expensive failures/serious vulnerabilities is an open issue.

On the tooling side many usual concerns arose, but there were also some specific recommendations to advance upon.

- Improving DCE Tool interoperability took effort in AQUAS and remaining interface incompatibilities need improvements..
- Achieving model compatibility or generality caused extra effort in order to cover, aspects like DCE traceability, schedulability or SW partitioning.
- Also, a more detailed documentation of modelling tools would be necessary to help developers

The use cases revealed most of the tool-associated challenges, but also recommendations for the general advancement of DCE.

- Certification in the light of DCE shall be treated in future research projects.
- Interference analysis by domain experts causes high initial effort, in particular for larger systems, and it requires experienced experts. The challenge is to adapt the approach so that it becomes more effective at finding the actual problems, reduce the “false positives”, and achieve scalability for larger and more complex systems..
- Investigations about additional aspects apart from those covered in AQUAS case studies (Safety, Security and Performance) should be treated in future projects

Several additional challenges were identified on a more general level:

- The integration of the human factor in dependability co-engineering,
- More wide-spread offerings for training, ideally dedicated DCE curricula at universities,
- Avoiding insufficient coverage or over-simplification when automating risk prioritisation,
- Sustainably influencing standards evolution across domains towards standardized DCE.

Change Control

Document History

Version	Date	Change History	Author(s)	Organisation(s)
0.1	18/02/2020	Draft table of contents	T. Gruber	AIT
1.0	07/06/2020	Final version	T. Gruber S. Mazzini J. Martinez M. Matschnig, B. Fischer P. Popov, L. Strigini P. Magnin, Q. Mankuni C. Robinson J. Cordero, P. Ruiz R. Ruiz J. Ouy I. Moreno K. Sharman J. Godot L. Aprvile M. Winkler J. Herter S. Li P. Mrnuštik	AIT Intecs Tecnalia SAG City SISW TRT Integrasys RGB ClearSy TASE ITI All4Tec MTTP AMT AbsInt CEA TP

Distribution List

Date	Issue	Group
08/06/2020	1.0	ECSEL JU all@aquas-project.eu

Table of Contents

1 Introduction 7
2 Context of this Document 9
2.1 Associated Projects and Roadmaps Before AQUAS 9
2.1.1 Projects Investigating Combined Techniques 9
2.1.2 Relation of AQUAS to the Challenges in the CPSoS Roadmap 10
2.1.3 Relation to the key recommendations from the roadmapping of Platforms4CPS 13
2.2 Relation to the Standardisation Report 14
3 Progress by Using AQUAS Methodology and Challenges 15
3.1 AQUAS Interaction Points 16
3.2 AQUAS Methods of combined analysis 19
3.3 Traceability of AQUAS IPs throughout the product lifecycle 19
3.4 Lessons learned in AQUAS 19
4 Progress and Challenges in the AQUAS Case Studies 19
4.1 Air Traffic Management Case Study 20
4.2 Medical Case Study 21
4.3 Railway Case Study 23
4.4 Industrial Automation Case Study 24
4.5 Space Case Study 26
5 Progress with AQUAS Co-Engineering Prototypes and Challenges 27
6 An Integrative R&D Approach: Conditions and Collaboration 27
7 Further Identified Challenges 30
7.1 Integration of the human factor in dependability co-engineering 30
7.2 More widespread offerings for training 31
7.3 Risk of prioritising automation over dependability 31
7.4 Standards evolutions towards standardised co-engineering 32
7.5 Enhancing policies in Europe/worldwide 32
7.6 Quantifying the benefits of DCE 33
8 Summary and Conclusions 34
9 References 35
10 Abbreviations 36
11 Glossary 38
12 Appendix 1: AQUAS Combined Analysis Methods 42
13 Appendix 2: Tools and Tool Combinations 45
14 Appendix 3: AQUAS Partners 51

1 Introduction

This report is an outcome of the AQUAS project, which developed a methodology and capabilities for co-engineering (CE) for the dependable systems industry. We call this dependability co-engineering (DCE). Both CE and DCE tend to be used synonymously in this document. The purpose of DCE is to resolve the problems arising from conflicts between safety, security and performance properties for the systems in order to avoid undesirable and critical consequences like injuries, fatalities or even catastrophes but also financial loss or excessive cost during the product lifecycle. The AQUAS approach consider advances for DCE methodology and considered the barriers that stand in the way of bringing optimisation and automation into mainstream practices. The applied co-engineering method was essentially based on *Interaction Points*, at which experts for the different qualities performed a trade-off analysis between safety, security and performance and tried at an early stage in the lifecycle to avoid costly iterations from later points in the lifecycle, saving cost and development time.

This document explains on the one hand the co-engineering related achievements accomplished during the AQUAS project and, on the other hand, points out what should be tackled in the future in order to penetrate industry with the novel approach and fully exploit its benefits. In this sense it gives guidance for future research programs and university curricula and can be the basis for establishing a roadmap for the future.

There are also challenges in standardisation concerning guidance for co-engineering, which need to be overcome; therefore, the following chapter explains first the difference between this report and the (also public) AQUAS report on standardisation. Then the AQUAS achievements are put in relation with roadmaps related to Cyber-Physical Systems (CPSoS and Platforms4CPS), and finally the connection of AQUAS with selected but important other co-engineering projects is explained. The relation to the AQUAS work packages is not given in detail as the respective confidential deliverables are not publicly available.

Chapter three elaborates on achievements in the AQUAS methodology, which is complemented by a table of combined co-engineering methods in Appendix 1, and the challenges which the methodology is still facing are explained.

The next main section describes the perspective of the five case studies, again providing information on what has been accomplished in AQUAS and what should be done in the future.

Chapter five explains first which enhancements have been implemented in tools and tool combinations in order to implement efficient co-engineering. Also here, Appendix 2 gives insight into details by listing examples of tools and their individual advancements in AQUAS, and finally the challenges in relation to tool development that are still to be tackled with in the future, are described.

The subsequent Chapter Six relates especially to the coordination and conditions that will be needed to support advancement of dependability co-engineering relating to impact and uptake.

After the previous sections have mainly presented the AQUAS work packages' point of view, the next chapter presents conditions and collaboration for an integrative R&D approach. Chapter 7 brings considerations about remaining important challenges for establishing co-engineering methodology: The human aspect in co-engineering, the need for appropriate training, the balance between high automation and the risk of incompleteness, the lack of co-engineering guidance in today's standardization, necessary enhancements in national and international policies, and the quantification of the benefits of co-engineering.

Finally, Chapter 8 presents a summary and conclusions.

For ease of reading, abbreviations, tool names mentioned in the text and DCE terms are explained in the sections 10/11 Abbreviations and Glossary.

2 Context of this Document

The AQUAS project follows three main goals:

- Co-engineering Goal
- Product lifecycle for Co-engineering
- Standards evolution for Co-engineering

This deliverable summarises the progress and open issues mainly in relation to the Co-engineering Goal but also to the Product lifecycle for Co-engineering goal. As explained further below, the progress and gaps related to the Standards evolution for Co-engineering Goal is described in a separate deliverable [reference].

2.1 Associated Projects and Roadmaps Before AQUAS

Already before AQUAS, co-engineering methodologies were attempted over a good number of years, but generally in isolated sectors and suffering difficulties with linking specialist domains. Even most research projects treated only safety or only security. Typically, separate engineering teams conducted separate processes for safety, security and performance. The specialists often restricted their thinking to their own domain of expertise, spoke only “their language”, and even refused to accept the arguments of other domain experts. This is known as separation of concerns and used to manage complexity (although it is also human nature to treat just their part!). Consequently, mutual influences were over-looked, redundant development took place. Due to late detection of contradictions, iterations across multiple lifecycle phases or systems were deployed with unresolved conflicts (e.g. security controls hampering safety) were often necessary and high additional costs occurred for the problem resolution.

2.1.1 Projects Investigating Combined Techniques

AQUAS was not the first project to deal with the interplay of safety and security. For instance, in the Arrowhead Artemis project (2013-2016), the mutual influence of safety and security was recognized and led to the development of the combined safety-security analysis method FMVEA (Failure Modes, Vulnerabilities and Effects Analysis) [10]. In the Artemis project EMC2 (2014-17) a combined safety-security development workflow was applied in an automotive use case demonstrating early how the interactions can be treated.

In particular, AQUAS inherits results of the project SeSaMo (2012-2015) to advance co-engineering. The SESAMO project first addressed the root causes of problems arising from the convergence of safety and security in embedded systems at architectural level, as the absence of a rigorous theoretical and practical understanding of safety and security feature interaction. Comparative review of relevant standards, identification of mechanisms for the design and development of safety/security critical systems, together with concepts and principles for risk management, analysis and assessment techniques and their interactions were “building blocks” for the elaboration of the SESAMO domain-independent general design methodology for the development of safety critical embedded systems, that was then experimented on industrial use cases in different domains.

Additionally experiences from MERgE (2012-2015) played a significant role in shaping AQUAS to investigate the uptake of co-engineering, where the safety-security design had been coordinated and investigated including a sizeable state of the art white paper report [9] .

AQUAS largely advances in the methodological approach by extending co-engineering to performance concerns, with use case driven activities spread across the product life, providing advanced interaction concepts and effective co-analysis, supported by a larger number of improved, extended and interoperable technologies.

A very interesting approach in the context of co-engineering was developed in the ECSEL RIA project AMASS, which ran mostly in parallel to AQUAS but started one year earlier. Its main focus was model-based multi-concern assurance.

AMASS built on the predecessor projects SafeCer and Opencoss. The Artemis projects pSafeCer (2012-14) and nSafeCer (2013-15), which had only safety in scope, investigated composable safety certification building on contracts between reusable components. The second predecessor project of AMASS was the FP7 project Opencoss; it focused on safety, too, and used a formal certification language to describe the dependencies in the safety case.

AMASS came up with a synthesis of the ideas of both projects to support constructing and maintaining a model-based assurance case taking care of interdependencies between, like in AQUAS, safety, security and performance. The assurance case was expressed graphically in goal structuring notation but with a transformation into an extension of the OMG SACM (Structured Assurance Case Metamodel). Features developed for safety in predecessor projects were extended to security and integrated in the AMASS metamodel, like for instance component contracts [3], variant management [4], or security-informed safety-oriented process lines [5].

In relation to AQUAS, the focus in AMASS was clearly on a comprehensive model to support the architecture-based generation of a safety/security/performance-oriented assurance case and its maintenance when the system model or the preconditions for it change. The interactions between the different qualities, in AMASS called concerns, were recognized but a workflow with defined Interaction Points was not a central part of the metamodel.

The AQUAS capability was significantly enhanced by all these projects, enabling it to extend to considerations for the full product lifecycle and to look at the challenges for uptake by industry,

2.1.2 Relation of AQUAS to the Challenges in the CPSoS Roadmap

There are many attempts at research roadmaps; the reviewers of the first AQUAS proposal submission recommended the one developed in the European Support Action “CPSoS – Towards a European Roadmap on Research and Innovation in Engineering and Management of Cyber-physical Systems of Systems” (https://www.cpsos.eu) as a benchmark. This roadmap essentially focuses on large scale systems whereas the AQUAS use cases are smaller and, due to the scale of the project, scalability was not investigated beyond what we tried out, so the roadmap is only partly applicable.

The CPSoS roadmap identified three core long-term research challenges and defined 11 medium-term research and innovation priorities

The three core long-term research challenges are

1. Distributed, reliable and efficient management of cyber-physical systems of systems,
2. Engineering support for the design-operation continuum of cyber-physical systems of systems, and
3. Towards cognitive cyber-physical systems of systems.

That shall be addressed in an inter-disciplinary manner and in collaboration of tool and solution providers, end-users, and research institutions.

When trying to map the three research challenges to the scope of AQUAS as defined in the FPP, it becomes clear that AQUAS mainly addresses challenge 2 because it supports S-P-S engineering along the entire product lifecycle. There is no strong focus on CPSoS (except in aspects of Case Study 1 ATM), and cognitive CPSoS in the sense of machine learning are not touched at all by the project. So we can identify core long-term research challenges 1 and 3 as open topics after AQUAS.

In the CPSoS support action, 11 medium-term research and innovation priorities were defined that should be considered and funded towards meeting the core challenges:

1. System Integration and Reconfiguration,
2. Resiliency in Large Systems,
3. Distributed Robust System-wide Optimization,
4. Data-based System Operation,
5. Predictive Maintenance for Improved Asset Management,
6. Overcoming the Modelling Bottleneck,
7. Humans in the Loop,
8. Integration of Control, Scheduling, Planning, and Demand-side Response in Industrial Production Systems,
9. New ICT Infrastructures for Adaptable, Resilient, and Reconfigurable Manufacturing Processes,
10. Multi-disciplinary, Multi-objective Optimization of Operations in Complex, Dynamic, 24/7 Systems, and
11. Safe, Secure and Trusted Autonomous Operations in Transportation and Logistics.

This list of research and innovation priorities is very heterogeneous. Part of them refers to a certain domain (e.g. Transportation and Logistics) or a system with specific functional requirements (e.g. Multi-disciplinary, Multi-objective Optimization of Operations in Complex, Dynamic, 24/7 Systems). Others are very general like “humans in the loop” or domain-independent (e.g. “Modelling Bottleneck”).

AQUAS, in contrast, is clearly a transversal approach, conceptually domain-independent and applicable to very different kinds of systems. Thus, AQUAS can be beneficial when it is used for developing methods according to the research and innovation priorities. The following table outlines this relation shortly.

Table 1 Relation of the medium-term research and innovation priorities to AQUAS

No	*Research and innovation priority*	*Relation to AQUAS*
1.	System Integration and Reconfiguration	As AQUAS treats the entire PLC, the phases are covered, although there is no specific focus on re-configuration.
2.	Resiliency in Large Systems	AQUAS methodology supports resiliency and some of the methods developed in the project demonstrated how a particular form of resilience, “proactive recovery” to deal with consequences of successful attacks, can be “engineered” by tuning the frequency of proactive recovery to achieve sufficient mitigation against even unknown attacks.

3.	Distributed Robust System-wide Optimization	AQUAS methodology, especially, the methods to help with the trade-offs resolution support robustness and seek optimization under the uncertainty caused by the limited knowledge about cyber threats.
4.	Data-based System Operation	This kind of systems have not been studied in AQUAS.
5.	Predictive Maintenance for Improved Asset Management	Predictive Maintenance was not a target of investigation in AQUAS.
6.	Overcoming the Modelling Bottleneck	The focus in AQUAS has been on using modelling for dependability co-engineering. In AQUAS the gap between models used for "system development" (such as SysML and UML) and models used for combined SSP analysis has been addressed. We have tools which allow one to switch seamlessly using advanced tool support from SysML to SSP models. These advances seem very relevant to overcoming the modelling bottleneck.
7.	Humans in the Loop	AQUAS co-engineering can support the integration of human factors into dependability engineering, e.g. by supporting combined analyses of the effects of human factors on interactions between safety and security. An example study on such a safety-security-usability interaction problem has indeed been developed in AQUAS. We identify this area as posing challenges on which much more activity is needed.
8.	Integration of Control, Scheduling, Planning, and Demand-side Response in Industrial Production Systems	AQUAS provides general co-engineering support for all PLC phases of such systems.
9.	New ICT Infrastructures for Adaptable, Resilient, and Reconfigurable Manufacturing Processes	The manufacturing domain was not treated in detail in AQUAS.
10.	Multi-disciplinary, Multi-objective Optimization of Operations in Complex, Dynamic, 24/7 Systems	AQUAS IS a project about multi-objective trade-offs. Resolving the trade-offs rationally (possibly looking for "optimality") is the essence of co-engineering. In many cases "optimality" is difficult to achieve, but an acceptable compromise is clearly of interest.
11.	Safe, Secure and Trusted Autonomous Operations in Transportation and Logistics.	The AQUAS methodology was validated, among others, in a railway domain case study; however the system considered was a platform door locking system and in this sense not a typical transportation system application. Nevertheless, the results obtained with the safe and secure railway door control system are transferable to Transportation

		Domain systems.

2.1.3 Relation to the key recommendations from the roadmapping of Platforms4CPS

Platforms4CPS provided a fusion and extension to roadmaps for cyber-physical systems (CPS). It was running in parallel with AQUAS and completed a year ago. There were 12 key recommendations out of over hundred identified. Many of the key points can be related to investigations that have also been underway in AQUAS. This is not surprising because CPS technology represents especially the integration challenges of safety-critical systems of which DCE is a prime enabler. Also of course DCE is an intersection for all technology. However, the points of higher relevance included:

Grand Challenge	Recommendation	Relevance with Dependability Co-Engineering
Research Challenges		
Trustworthy CPS for Autonomous and Smart AI – Societal Scale CPS	Develop a science of design for CPS with multiple links to application domains	Relevant in that DCE is an enabler here supporting fundamental multi-domain research. AI was mostly outside the scope of AQUAS but we did have a scouting market study: AI systems, which are increasingly being developed for tasks such as autonomous driving are unpredictable by nature. The classical methods for ensuring safety by verification and validation at design level do not apply to these.
CPS Edge Computing	Support research actions on edge computing algorithms and architectures	Relevant because it is a case of safety-critical systems connected to a secure cloud. Variations across SSP will need to be well understood.
Humans-in-the-Loop	Address the complex inter-actions between humans and systems with increasing autonomous functionality	Relevant and discussed later in this report.
Co-engineering of CPS system attributes	Advance techniques to man-age and automate traceability and trade-off optimisation between safety, security, performance and usability.	Highly relevant. This has been at the heart of AQUAS.
Innovation Challenges		
Defragmentation / Collaboration	Link existing activities to boost communication, avoid fragmentation and silos	Highly relevant. This has also been part of the AQUAS investigation relating to sustainable advancements and uptake related to coordinated collaboration.
Improve the uptake of technology by CPS industrial processes	Build supportive approaches to migrate existing industrial engineering processes allowing swifter time to market for technologies	Highly relevant and considered in AQUAS. Particularly for industry to evolve to some level of autonomy for SSP coupling. AQUAS has been considering a common technology foundation across industry (because already there is a need for mediation across standards) and also common baseline procedure to help industry adopt such technology considering come very different barriers to those faced by a component technology.
CPS Engineering, Interoperability, Complexity	Foster development of European tool chains for CPS	Relevant. DCE is a cornerstone for CPS. Without automation it represents a bottleneck to complexity.
Skills / Competence Provision for EU Competitiveness	Revitalise EU Engineering education and raise the status of engineering, embracing multi-disciplinarity ...	Within AQUAS we have learned there is a need for an integrative approach to R&D for integrative technology (as opposed to component or individual based). There are many more unknown technical challenges that arise requiring teamwork for solutions and standard project (component) approaches has required a steep learning curve for many – mechanisms need to be in place to support this.

2.2 Relation to the Standardisation Report

Standardisation is one of the most powerful tools of technological and economic infrastructures and greatly influences the competitive ability and the strategies of companies.

Therefore, it is as been important for all the AQUAS partners to recognise the benefits of standardisation for co-engineering purposes and to address findings that could improve the European and global framework of standards and maximise the uptake of the project results by Industrial, Regulatory, Academic and other stakeholders. Particularly in relation to mediation between e.g. a safety standard and a security standard.

Deliverable D1.9 “Report on the Evolution of CE Standards” [1] provides an overview of existing standards not only in the context of AQUAS use cases, but also addresses other relevant approaches being pursued by standards developing organisations in other domains, as well as modelling and tool interoperability standards and emerging frameworks that are relevant for the AQUAS methodological and tool development approach.

The interplay between SSP dependability attributes is being increasingly accepted by 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, or even lacking (as in the case of performance).

Even if AQUAS does not have a use case in the automotive sector, we observed there is vigorous ongoing standardisation activity: of particular relevance to AQUAS is the remarkable fact that the current automotive standardisation activity involves three standards in evolution) that address exactly all the **three dimensions** treated by AQUAS, safety, cybersecurity and performance, and require interactions between these properties. AQUAS partners have established collaborations for the analysis of co-engineering in representative standards. Based on the involvement of AQUAS partners in standardization activities, or the establishment of contacts with relevant stakeholders in the bodies, or even dissemination through public events, the identified gaps, needs for the identification of SSP interactions and analysis cooperation have been promoted with the purpose to build consensus and actively influence the evolution of the development processes supported by the standards.

The collaboration effort described in this document provides a number of helpful results, but also foundations and clear directions for standardisation and alignment between the outcomes of the project and standards in the near future.

A major result of these activities has been the collection among AQUAS partners and the submission of requirements for the MARTE 2.0 OMG standard that is currently underway for a major revision, with concrete plans to submit a concrete revision proposal.

The principal requirements 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 in particular modelling extension for dependability, safety, and security. Note that part of these extensions are applied in the AQUAS method for dependability modelling and combined analysis using stochastic activity networks.

Another recent success not reported in [1] has been the contribution to IEC 61508-3 and IEC 61508-1/2, obtaining that the consideration of cybersecurity during Risk and Hazard Analysis Phase is now a normative requirement, with the necessary follow-up processes if an impact of security threats on safety is identified.

3 Progress by Using AQUAS Methodology and Challenges

Co-engineering with interaction points (IPs) has been central for AQUAS, a particular form of "co-engineering" of qualities of critical embedded computer-based systems deployed in *untrusted environment*.

IPs are points in the product lifecycle, when non-functional properties of safety, security, performance (SSP) are analysed *together*. These analyses allow developers to identify potential conflicts between safety, security and performance of the system under development, conduct a trade-off analysis and seek to find an acceptable compromise between the conflicting properties of the particular system; or to identify synergies between design features intended for these different goals, and thus design simplifications or other improvements.

The AQUAS methodology delivers improvements in:

- *System quality* by applying a *holistic approach* to analysis of systems by applying different methods for combined analysis at different stages of the product lifecycle. Such an approach allowed us to reduce the risk of omitting subtle problems due to the interdependencies between safety, security and performance. The models of combined analysis differ in the level of detail and knowledge about the system under development available at different stages of the product lifecycle, with complexity and level of sophistication of the analyses increasing throughout the product (development) lifecycle – from methods which help identify conflicts between different requirements and find acceptable resolutions via analysis applied in design to help explore rationally the design space available for a particular system, and eventually in setting complex V&V scenarios to check system behaviour under combination and accidental and malicious threats, likely to occur in practice.
- Building systems more *cost-effectively*. Early resolution of conflicts between requirements is seen as a major advance of the state-of-the-art, which promises a reduction of the overall cost of the developed systems. This includes the cost of development, of course, but also the cost of maintaining the systems after their deployment. Measuring the actual cost reduction is not feasible in a single research project, limited, in the case of AQUAS, to development stages in a product's lifecycle; and no reliable way to measure the savings in operation by avoiding expensive failures/serious vulnerabilities was available. This difficulty was addressed by creating a "cost model", following the tradition in software engineering, to help with the estimation of the costs and likely savings in operation.

The rest of this chapter is structured as follows:

- We provide an illustration of how IPs fit in a typical lifecycle model (section 3.1)
- We illustrate what IPs are made of (section 3.2). We provide a list of the methods for combined analysis in which at least two of the SSP properties are addressed, in Appendix 1.
- We illustrate the relationships between IPs and the respective entities that they consist of and how they are traced throughout the lifecycle (section 3.3)
- We summarise "lessons learned" from applying AQUAS methodology in practice (section 3.4).

3.1 AQUAS Interaction Points

In AQUAS, IPs were defined as points in the product lifecycle, when non-functional requirements of safety, security, performance (SSP) are analysed *simultaneously* by applying suitable methods of *combined analysis* which seeks to establish whether the system's non-functional properties meet their respective requirements. These analyses allow developers to identify potential conflicts *early in the lifecycle*, scope the space of trade-offs between the attributes of interest, and check if an acceptable compromise can be found between these properties for the particular system under development.

An activity diagram is shown in Figure 1, which illustrates a typical PLC and covers all phases of a typical product development life-cycle (requirements engineering, system design, implementation) and operation. While the IP in development occur sequentially with clear dependencies between the phases – the artefacts a particular phase delivers as an output (e.g. a requirements specification or a design documentation, etc., eventually quantified relations in a database) are used as inputs to the next phase, an IP in operation is somewhat different, in that it is normally triggered by a "change request". This may be a result of observing a deficiency in operation (a critical fault or a vulnerability). Responding to the "change request" requires an IP to establish, by a suitably conducted IP combined analyses, that the changed system satisfies all requirements and is also free from the observed anomaly (due to a design flaw or to a discovered vulnerability).

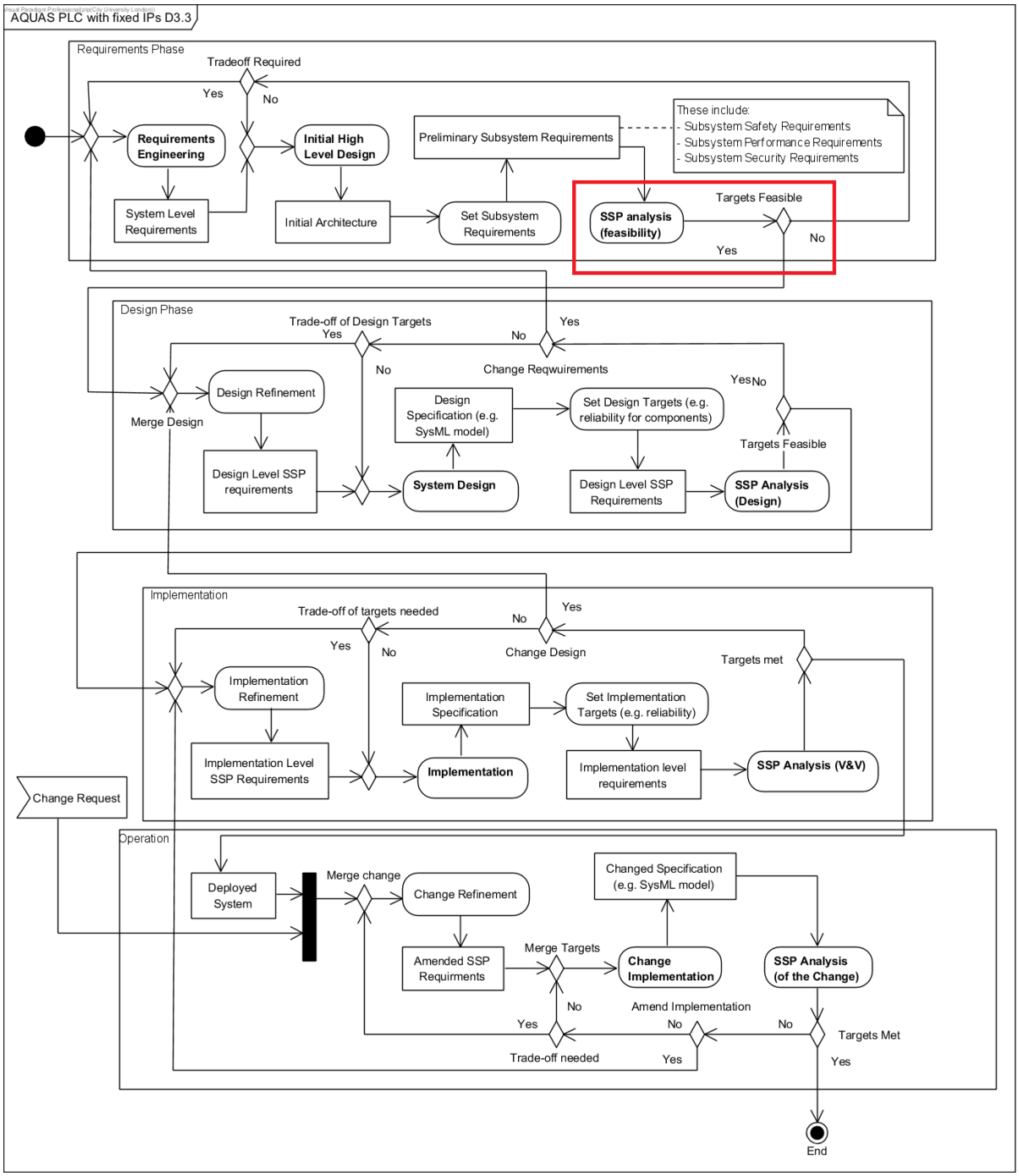

Figure 1. Interaction points embedded in a PLC.

The IPs, labelled in the diagram as “SSP analysis” (e.g. the activity highlighted in red in Figure 1) consists of one or more combined analyses, as illustrated in Figure 2.

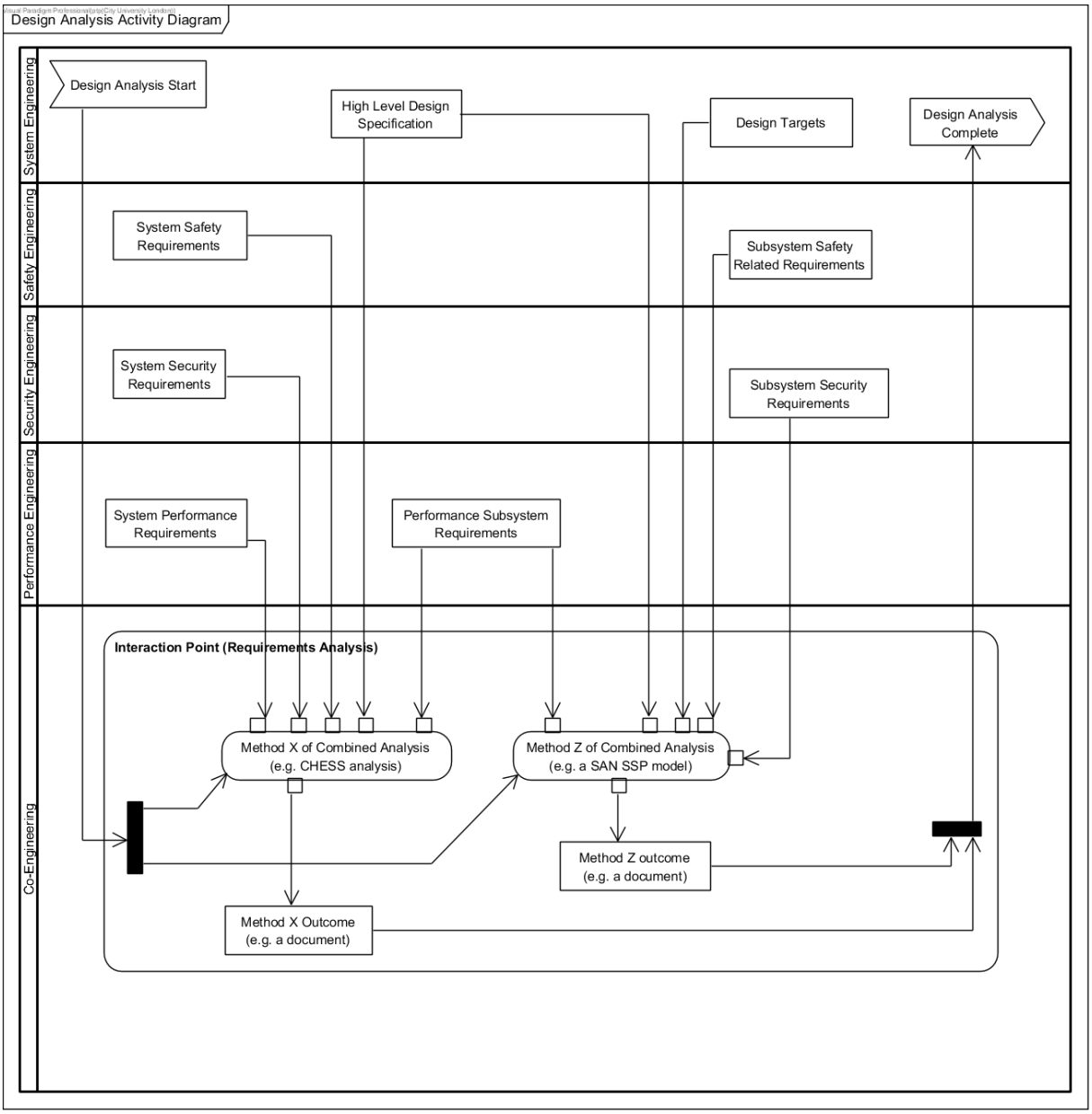

Figure 2. An IP at Design phase of PLC.

Let us look at the IP at design phase first, one of the IP described in D3.2. For a given design, e.g. captured in a SysML model, a number of combined analyses can be undertaken, which look holistically at different combinations of non-functional properties. The example shown in Figure 2 refers to two of the combined analyses used in AQUAS: the analyses possible with CHESS (e.g. dependability analysis and the Worst Case Execution Time (WCET) analysis and the combined analysis using stochastic activity networks (SAN) formalism. Each of these analyses takes as *inputs* a set of artefacts (modelled as objects in the activity diagram) which are created in the other lanes of the diagram. Among them is a model of the system architecture (e.g. as a SysML model), created in the System Engineering lane, or some of the system requirements which are captured in the Safety Engineering, Security Engineering or Performance Engineering lanes, respectively. Each of the combined analyses, in turn, produces one or more *outputs*. Only when the outputs from all combined analyses are produced is the IP complete and the outputs from these combined analyses are passed as an output from the respective IP. These will then be used by experts to check, if the non-functional properties meet their respective non-functional requirements.

3.2 AQUAS Methods of combined analysis

A large number of methods for combined analysis have been used in AQUAS. These are summarised in Appendix 1. In addition, a large set of analysis methods dealing with a single non-functional property - either safety or security or performance – were used in the project, supported by software tools.

3.3 Traceability of AQUAS IPs throughout the product lifecycle

Depending on PLC and the specifics of the projects, IPs can be predefined (i.e. before the start of a project) or may be used when the need for IP occurs, e.g. a “change request” is made in operation after a serious failure or a near miss or after a serious vulnerability has been discovered.

Managing the process of co-engineering with IPs requires that an efficient way of tracing IPs be deployed, which should allow for recording the combined analyses used in different IPs, together with the inputs they depend upon and the outputs each combined analysis produces together with the tool support used in analyses.

Such tool support was developed in AQUAS – a software tool by Magillem was successfully trialled throughout in the Industrial Automation Case Study.

3.4 Lessons learned in AQUAS

The AQUAS methodology has been developed and assessed by applying it in practice on five case studies. Each case study applied a subset of analysis methods and tools that seemed to fit their need and organised their application into IPs. The conclusions reached by the leaders of these case studies (industrial companies applying the methods to their development projects) were collected. Important observations included:

- interaction points with combined analyses were judged applicable in practice
- pre-planned IPs were helpful for searching and resolving early in the lifecycle problems that could arise from interactions between SSP-related events and design precautions, which would otherwise prove costly to tackle at later stages
- the combined analysis techniques trialled appeared effective and useful. The potential for fast adoption varied depending on the maturity of supporting tools
- interoperability of tools so that a toolchain could support a more extensive combined analysis was a plus
- the changes needed to incorporate the IP concept and the specific analyses in an existing PLC could be made affordable with adequate tool support.

4 Progress and Challenges in the AQUAS Case Studies

The activities taken in AQUAS resulted in the evolution of methods and refinement and feature extensions of tools for dependability co-engineering. Within this chapter, the major advancements gained from methods and tools in AQUAS for each domain’s case study are listed and described briefly. Even though many co-engineering challenges have been approached successfully, there are still issues that need to be solved in order to fully deploy AQUAS processes and achieve the cost reduction and the gain in dependability and efficiency. These future challenges are also presented in this section to complete the picture of achievements and needs in the AQUAS Case Studies.

4.1 Air Traffic Management Case Study

The AQUAS approach brings in to Integrasys' baseline a new way to consider together the S-S-P requirements through the all PLC and, in particular from early stages, including interaction points and combined analysis supported by tools. For Integrasys as a small company, this project is defining the new procedures applied to our PLC, because previously to AQUAS, the safety, security, and performance metrics was assessed separately and many of the decisions were based on experience and not as now based on combined analysis results.

The results and conclusions show that the introduction of AQUAS methodology in the current work methodology can reduce the efforts and improve the quality of our development process. As a new methodology, it has required a significant effort and learning curve for implementation in the current baseline. Moreover, we have also estimated a large benefit in the operational and maintenance phase of our products: the thorough analysis in early phase, direct impact in quality of the product, and more importantly, ease the adaptation of any of our development solution to customer/market/regulations. So, the impact of any customisation of a solution that would be easily traceable and any impact of performance/security can be previewed before starting the implementation.

One of the strengths to be highlighted has been the detection of conflicts or interferences at concept phase, allowing them to be addressed at design phase (although the modelling/analysis becomes more complex), reducing the efforts in testing and validation phases and decrease costs due to the nature of this domain.

Regarding previous working methodology, we highlight the next added AQUAS features that we have acquired:

- Common data and artefacts representation across all PLC stages, with semi-automatic readjustment to issues or bugs found in testing phase.
- Definition of security and testing scenarios and its integration in a security requirement management tool for security traceability and management through overall PLC stages.
- Carried out at design phase schedulability and WCET (worst-case execution time) analysis with realistic platform modelling (platform selection).
- Analysis of platform partition and resource allocation in order to evaluate performance and security requirements in order to find better allocation/binding solution and to select the best candidate platform.
- Combined analysis of security and performance based on a probabilistic model, which is built using the Stochastic Activity Networks (SAN).
- Generate input data for unit/system testing in order to check the robustness (performance is fulfilled or not) of unit/testing regarding disturbances.
- Confirm the absence at system-level of dead-locks and live-locks and verify safety and security properties of source code.
- Analysing multi-threaded C/C++ programs on the binary level detection, providing a monitoring layer offering notification about important events in execution, such as thread synchronisation or memory accesses.

However, although the AQUAS methodology and the obtained results are excellent, more efforts should be focused on increasing the co-engineering and integration between tools in order to

facilitate the work and reduce time and efforts of engineers, developers or designers. Next, we comment on some limitations or gaps to be solved in the future or advancement with ongoing work:

- As a general issue, increase the integration between tools in order to reduce time and efforts, and, more specially, tools from different PLC phases in order to reuse material. Examples:
    - The importance of the on-going collaboration between City and Intecs is to derive a SAN model from a system model developed in CHESS (as is the case for this use case). Even a partial success of integration will drastically reduce the effort required to build a SAN model.
    - Integration of security requirement management tool with other tools of PLC chain in order to make automatic the security traceability and management process.
- For hardware design, the work should be routed to collaborate with other tools (e.g. CHESS) for schedulability analysis and SW partitioning in order to decrease design time while improving system implementation reliability. Verification and validation activities on a real board environment are needed for different aspect, from methodology refinement to system components improvement, while the Design Space Exploration can help us to guarantee the fulfilment of input requirements.
- In verification phase for the test case generation in MoMuT, much effort was needed in order to implement a test interface in the ATM application and to generate the disturbance model (several iterations). A solution could be providing more information in CHESS models in order to reuse these models in the verification tool (self-contained), at least to generate preliminary test cases.

4.2 Medical Case Study

Here, the baseline situation before AQUAS, was as follows: The dependability co-engineering capabilities for Medical Device applications were mostly limited by manual co-engineering of safety and performance with little consideration to security requirements. Initially, co-engineering was carried out with a “very basic” interference analysis method implemented in Excel sheets, manually reviewed for coherence analysis.

In addition to safety and performance, the quality system attribute “performance” (timing) was analysed by a combination of several tools, namely IBM Rational DOORS for the requirement´s definition and system architecture and fault injection techniques.

The Product Lifecycle already had concepts for performance/safety co-engineering for various phases of the PLC, using the IEC 60601 standard on Medical electrical equipment; security was not considered at all, neither the norms did address the issue.

As baseline, the resources devoted to each stage of the PLC were estimated theoretically, but some adjustments had been done since the company had developed the BARICONTROL device in the 90´s, and by that time there were considerable differences in the development culture, and regulatory requirements. For example, security was not an issue. Therefore, it is difficult to make a good comparison, in terms e.g. of development process costs, because current security requirements mean larger development cycles; Besides, compliance with regulation was much easier to achieve.

Before AQUAS, requirements for product qualities were captured inhouse with requirements engineering toolset (WORDS e.g. for safety, performance and functional requirements engineering). However, automated linkage of requirements to the succeeding Design phase was hardly possible. Furthermore, security requirements gathered during security analysis were stored in spreadsheets,

again without automated linkage to succeeding PLC phases. When it comes to Design and Implementation, functional specification and design specification were manifested as text. Test plans and Test reports (e.g. coverage reports) were mostly also text format files (stored in a database).

In summary, the safety, security and performance approach was based on the decisions of very expert professionals, but not following a methodology nor using tools for combined analysis.

The AQUAS Medical Devices Use Case has incorporated the IP strategy in the consideration of SSP requirements, design and verification phases of the product life cycle. During the demonstrator implementation, advancements have been evaluated in each of the stages, as well as the investment needed for these advancements.

The evaluation of the demonstrator shows that some efforts could be moved from the verification phase towards the early requirement and design phases: The Requirements tasks is larger due to higher need of interference analysis and combined analysis, particularly also because security has brought about new activities which were not used in the past. However, this also brings higher savings in manpower for the Implementation phases and Verification.

AQUAS seems to be a sound methodology because dependencies that cause expensive design iterations are discovered at earlier, in the Requirements Phase, instead of late, in the phases V&V and Operation. The AQUAS toolset provides a lot of improvement in the capabilities for tracing SSP attributes across PLC phases.

Within AQUAS, several tools and methods have shown capabilities improving the above-mentioned situation:

- A hazard and operability (HAZOP) analysis, for both safety and security aspects, was run, on a specific use scenario: closed-loop control of blood pressure during a surgical intervention.
- Safety and security assurance of AQUAS medical use case with OpenCert, as an introduction to the demo on the Infusion Pump assurance case illustrative example using a new version of Eclipse OpenCert. CHESS, CHESS is a systems modelling tool based on Papyrus UML+SysML+MARTE. It was used in this case study to model the architecture of the RGB product, TOF CUff.
    - SafetyArchitect is a safety risk analysis tool that works at model level. It can create Fault trees and FMEA/FMECA tables. Modelling the TOF Cuff system in Safety Architect tool can be done at functional and hardware levels.
    - CyberArchitect, a tool for security analysis, to provide S-S co-analysis.
    - OpenCert as a tool to support the certification of critical systems. It helps to manage assurance projects, argumentations and the body of evidence to demonstrate compliance to standards.
- In order to get a code analysable by Frama-C plug-ins, macros have been used allowing the use of a single source code targetable for both the Renesas compiler - TOF CUff Microcontroller - and the Frama-C platform.
- Tools for Combined safety and security analysis including asset identification have been developed. These include Safety-hazard analysis; Security-Risk assessment; Security – derivation of attacks; Security-threat analysis; Cause-effect relations shown between attacks, threats and hazards.
- Authentication works on Security: The authentication work has been a joint AQUAS task, in which a probabilistic model has been created to ease comparison of alternative authentication protocols (e.g. before each critical command vs periodically vs continuous).

- Cost Analysis: AQUAS has analysed data from RGB PLC development processes, producing an estimate of the cost difference between the PLC before and after the introduction of AQUAS processes.

Before AQUAS, security was not an issue. There was no need to detect quality intersections by systematic analysis; In AQUAS, tools such as SafetyArchitect /CyberArchitect of All4Tec or medini are specialized for detecting such intersection. medini can be applied for the analysis and detection of safety/security/performance dependencies. A2K was applied in the verification PLC, so interdependencies were analysed for performance, and safety while lesser effort was put into security. medini analyze's capabilities were focused primarily towards capturing safety/security requirements and their dependencies; It proved a useful tool to provide assistance for finding interdependencies between quality attributes and providing a base for quality attribute experts to discuss and decide when and how to analyse these dependencies.

It must be considered that a thorough analysis of a complete system, is a very demanding task, requiring a lot of man months dedication with expert involvement. With the newly introduced tool-assisted interferences analysis, it is now more probable to discover dependencies in an earlier phase (requirements phase) than it was before. In some situations, it is helpful to reduce the number of dependencies by abstracting requirements into requirement groups. Such technique should only be applied on non-critical requirements, but for all others the effort for interference analysis is reduced selectively. Another positive effect of the latter is that costly dependencies are now less likely to reveal themselves late in the verification phase, operation phase or implementation phase, but are rather caught earlier, in the requirements phase.

Experts estimate that the percentage of intersections discovered within the requirements phase is higher and within the Verification and Validation phase is lower with AQUAS methods.

As for limitations, it would be important to devote efforts to interoperability between tools so that the work can be easier and the required resources in terms of time and effort are less.

Also, in AQUAS UC2 Medical Application, certification is out of the project time scope, but it is always an issue, that must be taken into consideration for reaching a final trade-off decision.

4.3 Railway Case Study

CLEARSY expected to discover the most efficient way to introduce security in the existing product life cycle. During AQUAS, CLEARSY gained experience and a methodology in achieving this goal. With the concepts of Interaction points and Interference Analysis, the risk that comes with the introduction of new requirements is quantified and controlled.

To achieve the introduction of security engineering, we defined a safety/security interaction point concept. This IP was accompanied by special co-engineering meetings during which were discovered synergies between functions from safety and security requirements. This has been further analysed and a joint function has been designed to cover both requirements. Without this Interaction Point, we can imagine that safety and security experts both would have defined their own functions and transmitted them to the system engineer. In the best case the system engineer would have identified the synergy and reacted, a few days would have been wasted. In the worst case, both functions would have been developed independently and we would have discovered during the tests that they cannot run in the same time. Weeks or months would have been wasted.

The interaction point at concept phase also rose a risk in performance. Safety experts had experience in embedded development and could warn the security experts about a performance risk with a particular function on the target microcontroller. The Interaction Point can help transmitting knowledge between departments.

To mitigate this risk, CLEARSY experimented modelling and simulating with new tools. With TTOOL from MTTP, we explored the possibility of a performance model, with realistic time constraint. With Amesim from SIEMENS, we modelled the environment of the system with continuous time and physics. By connecting the two models we were able to solve the safety/security and security/performance problems that came from Interaction Points.

This will be of a great help for all embedded system development that can come in the future and may require simulation and load estimation.

For CLEARSY, it appeared that the interaction points are an efficient tool to materialise expectation and keep note of the moment decisions were taken. In particular, it becomes possible to represent the evolution of the project as a journey between interaction points, and better trace the consequences of actions and decisions taken during meetings. This eases the writing of Post-mortem documentation. In the context of safety systems, the points contribute to improving the certification process.

On the challenge and limitation side, some points can be noted that concern the adoption of the AQUAS product life cycle by a company like CLEARSY.

The first one is about self-confidence and IPs. As an SME, CLEARSY works with small groups of people that know each other and design phases of new products can be very informal, starting from casual conversations that slowly transform into project meetings. This agility does not seem to be compatible with the presence of an external expert, the security expert in our case. Moreover, at the beginning of such projects, lots of points are open or approximative and it might be difficult to share information about the project in this state with this external expert. We may not have answers to all his questions and more generally, it's difficult to show an incomplete and imperfect project to a stranger. On the other hand, defining the IPs is a good opportunity to define the scope and the boundaries of the project and take a good start.

The second point is about the additional cost of the IP and Interference Analysis. During the AQUAS development of Coppilot, we demonstrated that additional models required development time that was compensated by a cost reduction in analysis and global development. Yet, this cost needs to be accounted in proposals and detailed in costing. AQUAS gave us an example but not a generality, some work and experience are still required be able to forecast our need for additional models and Interaction Points in future product development.

4.4 Industrial Automation Case Study

Starting from a generalised basic PLC, tools and methods for developing Industrial Drives, several advancements towards safer and more secure product development was achieved. The previous statement is supported by the following arguments supported by the advancements and novelties made in methodology and tooling:

- Higher confidence in system architecture and complete and consistent set of requirements with a novel approach for Interference Analysis applied in the Concept Phase. Having the ability to find interferences with automated tool-support (medini analyze) eases the process of interference analysis sessions, where experts (system architects, security experts, safety experts, etc.) can judge if a potential interference is in fact one that should be analysed. Furthermore, with this approach, experts decide when and how to analyse interferences – thus also process management is supported by planning in advance when to analyse interferences. Interference Analysis is carried out very early in the PLC and when this is done thoroughly, conflicts are analysed, solved and quality attributes balanced. In whatever magnitude Interference Analysis is executed, the margin of unknown surprises that might show up later is reduced and thus costs are saved.

- Simulation-based assessment of the current system architecture with Stochastic Activity Networks allows the retrieval of statements about the system, especially safety and security statements. Several security attacks and their effects on system safety (dependability) can be carried out in early PLC phases. Simulation results are used to define safety and security measures that might have been missed otherwise such as a safety feature for cleansing the client application in the Industrial Drives use case.
- An important aspect nowadays is the analysis of security and its effects on performance, as well as security verification itself. These open questions were tackled by two distinct methods/tools for security/performance combined analysis. First, TTool, in combination of ProVerif for formal security verification, enables the analysis of the system architecture in combination with timing constraints. Valuable information is gained by exercising different security mechanism, e.g. different keys for encryption and authentication and their implications on performance. Applied early in the PLC, such information is suited to support choosing right security measures together with sufficient hardware and thus saving costs and efforts upfront. Second, the tool ThreatGet enables the simulation of system architecture with annotated timing and security properties. Similar to the approach with TTool, the methods it applies lead to statements about the system security, e.g. pointing out the portions of the architecture where security and timing violations happen. The process of picking the right security countermeasures is eased with this method.
- The ability to handle and link artefacts throughout the whole PLC was enabled by the Magillem Content Publisher, which makes the whole product development process easier by making information easily accessible for all stakeholders (some examples are system architects, developers, safety engineers, safety assessors, security engineers, etc.) in the project. This tool plays a central role in system development and information management since it allows linking any artefact such as requirements, code and system architecture components.
- Virtual prototyping of the complete Industrial Drives Application enabled creating a digital twin of a motor control application. This was achieved by the orchestration of several tools, SystemC (for discrete logic models), AMESim (for motor models) and QEMU (for hardware/processor models). Concerning co-engineering with respect to performance and security, this digital twin's benefit is that it enabled security/performance verification with a real-world security test on off-the-shelf hardware. Security weaknesses could be identified, and the system concept changed accordingly. Without a virtual prototype, this would not have been possible in such short time – instead, a physical demonstrator would have been needed, which would have induced delays in the PLC.

Even though there are some advantages, as stated above, some of them are no freebies, but come at a certain cost – there are still open points in the tools and methods refined during AQUAS:

- Interference Analysis stands out as a systematic approach for handling all kinds of dependencies between system quality attribute requirements and architecture. However, the complexity of systems and the high number of various requirements result in huge sets of potential interferences. Analysis of these sets by domain experts requires high initial effort. That challenge was tackled by abstracting requirements – but still, this reduction of the number of potential interferences that must be analysed might not suffice. Additionally, abstraction must be done carefully, best by senior experts having a lot of experience for the products at hand. Sloppy, or erroneous, abstraction increases the odds in favour of missing important details that would have led to an interference with detailed analysis. With current knowledge, this approach should only be applied to small systems or sub-systems.

- Improvements for the digital twin could be the usage of real CPU models (those used in QEMU have different timings). However, a do-over with real CPU models would slow down the simulations significantly. Thus, such an approach is only reasonable if very powerful machines are available.

4.5 Space Case Study

In the space domain, the current baseline approach, before AQUAS, concerning the handling of different quality attributes, in this case SSP concerns, consists in treating individually and sequentially each aspect, giving priority to safety, and then trading-off the performance and security to reach an acceptable requirements compliance. The introduction of multi-core architectures on board has given the opportunity of improving the functionality and performance of the products, at the cost of adding complexity on the cores' utilisation, because of the derived safety issues. This scenario motivates AQUAS methods and tools, with the particularity of involving coordinated work among different expert teams, against individual work already mentioned.

By using AQUAS procedures, important cost savings have been demonstrated to be achievable by:

- Introduction of the interference concept (from the concept phase), through convenient tagging of requirements and identification of candidate interferences.
- Improvement of requirements (compliance) traceability, with the help of tools allowing partial integration among themselves.
- Use of formal methods to reduce rigorous testing, replacing it with automatic testing tools. Concretely, using design tools supporting schedulability analysis and early validation/verification.
- Specific tools on the multicore domain (such as TRT's uMetrics approach for WCET analysis)

Some remaining challenges to be approached in the future for this use case are:

- The lack of standardisation of co-engineering processes: on one hand, there are efforts on-going to introduce security in the ECSS-E-ST-40 SW engineering standard, however this is an incremental approach rather than a CE approach; an initial gap analysis of the standard, with recommendations on how to map UC5 activities to standard life-cycle has been made in AQUAS.
- Usage of automatic code generation methods. For larger multi-core systems, it would be desirable having code automation techniques to lighten the code implementation process. This has been partly considered in AQUAS UC5 with some "out of the box" code to be used (decision taken at the concept phase) and with the hypervisors option.
- Extending co-analysis activities beyond verification phase. In AQUAS UC5, CE activities are focused on requirements, design and verification stages, and do not cover implementation, nor maintenance phases.
- Focusing on additional aspects apart from SSP (Safety, Security and Performance).

5 Progress with AQUAS Co-Engineering Prototypes and Challenges

Throughout the AQUAS project, different tools were developed to support all aspects of co-engineering. At the end of it, several successes were achieved, but some challenges still remain.

The AQUAS methodology is the base for improvements and the evolvement of tools that hold co-engineering as their core ideal.

One of the first **successes** is the ability of the providers to combine their tools to tackle issues in some UC. Especially, the interoperability and interfacing between tools towards a seamless design flow was enhanced. For example, the toolchain made of medini analyze, CHESS, and Mobius enables the design flow starting from requirements and architecture modelled in SysML, system decomposition (software/hardware) with timing analysis, safety and security annotations in CHESS, and finally stochastic analysis network-based analysis with Mobius. Latter toolchain poses a systematic, model-based way for system design very early in the PLC.

Another **achievement** that emerged during the project are the improvements that were made by the tool providers for their original tool. Co-engineering tool capabilities were enhanced and aligned to the AQUAS methodology. As an example, the basic concept of safety-security-performance interference analysis lead to the implementation of novel features in the tool medini analyze (automation-support for requirement interferences, also with standard requirement catalogues).

On the topic of **challenges**, a major one is that new tool features were initially designed for a specific use case where their underlying methodology is usually generically applicable, however, building tools fit for usage in multiple domains is a specific challenge that requires further tool extensions and enhancements. To conclude, some areas still need to be invested in to facilitate the co-engineering and the communication between tools – enhancements on that can only come by their application on many use cases (evolution of software).

For further information, a list of tools, their co-engineering functionality and open challenges together with plans for their future is given in Appendix 2.

The tools providers kept track of the TRL of their tool for the duration of the project, and we look at an average value to estimate the overall progress of the tool's features.

At the start of the project, we had an overall TRL value of 3 which means that most tools were still in a research stage to prove feasibility. By the end of the project, we reach an overall TRL value of 5 which represent the end of the technology development for most tools and even the beginning of a demonstration of said technology for some.

If we wanted to think about enhancing the tools even further and maybe reach a higher TRL number we would need to think about the type of use cases that we would integrate in the project to begin with. Tool providers have implemented DCE functionalities restricted to the PLC phases addressed in the use cases. They identified the need to have more industrial use cases, which would enable the tool providers to invest more time and more effort in the development, to achieve a higher coverage of PLC phases, and also to go further than the proof of concept.

6 An Integrative R&D Approach: Conditions and Collaboration

Considering the engineering of a cyber-physical system (CPS), such as air traffic control, helps to put things in perspective. Figure 4 shows an overview in relation to developing a CPS with key characteristics shown on the initial level, followed by many layers of integration of which a few are shown on the diagram. DCE or coupled dependability is right at the top of these levels of integration.

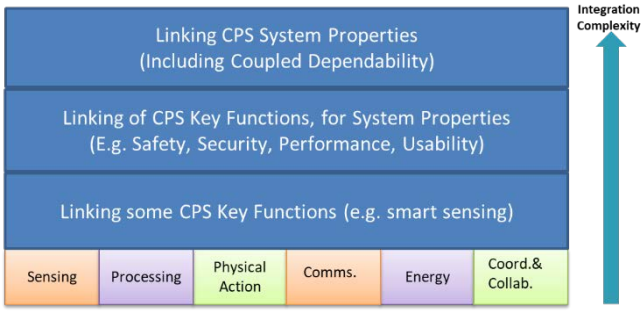

Figure 3: Caracterisation of a CPS and some integration levels.

What does this mean? Well there are some key differences between the lower level and the top level. For instance we are moving from what may be a single problem solved by a few people, to what is a multi-faceted problem requiring a much larger group to solve. DCE has the capacity to influence all the technology in a system and is the most influenced by industrial processes and national policies. This also makes DCE a type of *destination technology* rather than a contributive/component technology – the component technologies pass through DCE and other integrative technologies in order to be used. A very important implication here is that DCE (and integrative approaches) can make the uptake of many other technologies easier. Also in the higher layers the uptake of integrative technologies have many specific barriers causing the timescale to be much longer. In industry there is a tendency on individual levels to see long term advancement as lower priority, especially if they often have work due yesterday, which means advances at higher levels get put off - even when it means future reduced pressures and higher profitability. To have good advancement also means building momentum with the stakeholders which takes time and if it stops it is difficult to restart. Basically this is a very good example of challenges and support related to sustainability.

Related Challenges/Recommendations that can be provided here:

- The **Integrative Approach** should have prominence in DCE projects. Mechanisms to support this need are to be developed. This is a mindset or culture needed for these types of projects. New competencies are needed for coordinating and contributing to this high level integration.
- **Providing visibility of progress and advancement.** Structuring to enable partners to show how they fit in. Also metrics provided some good support. In AQUAS we broke down all 13 objectives into achievable actions, usually one action assigned to a partner. These were quite extensive excel sheets indicating what had been done for an action but complex to present – so we had equivalents simply showing colour representation for advancement. This provided good progress indicators, though we had some challenges.
- **Aligning on common foundations**. A company could take their own direction for automating DCE, in fact this is very likely. However it makes sense to have at least a common foundation that is built upon - because all safety-critical systems have safety and security certification to adhere to as well as other standards. At least at this level there should be the ability for automatic mediation between these. Furthermore advancing automation of DCE moves closer to self-embedding components or technology – that is capability to provide some feedback about their impact to dependability of a system. Key questions here include:
    - Describing the relationships (e.g. impact of changing a safety trait with a security relation).
    - Mediation between different safety/security ontologies.

- Automatically adjusting interrelated SSP at the operation stage for an adapting system.
- Depth of interrelations and interrelations across the product lifecycle needed to be considered.
- Length of time interdependencies can be left untreated in development (technical debt).
- Extending the community.

- Above-project level:
  - **A PPP Centralised Technical Coordination Body.**
    - The domain suffers from scope explosion with impact dilution.
    - Metrics to show how funds support high-level advancement.
    - Continuity, transfer of assets\benefits between projects including lessons learned and collaboration mechanisms. Interruptions have biggest impact at higher integration levels.
    - Proliferation of projects managed.
  - **Engaging with policy makers and regulation.** Especially efforts to support industrial competitivity on long term factors (system development is long-term, but is also a gateway that can increase general uptake of technologies).
  - **Support tools from funding bodies supporting "integrative approaches".** Particularly group problem – group solution tools. For instance DCE is about as multi-disciplinary as it gets. A reference glossary is important for a common understanding (the AQUAS one is later in this document). Use of the glossary can be another challenge, but at least it can resolve interpretations quickly.
  - **Public SSP trade-off examples database**.
  - Building a list of **recommended reading material**. Votes of confidence by recognised specialists would be useful. This links with the recommendation on taught courses elsewhere in this document.
  - Generally **more and focused funding** for system engineering. It is this level that supports a higher transfer and use of other technologies. Appropriate metrics to be established, but this should be (perhaps can only be) iteratively improved by application. So good metrics are unlikely to arise without good investment.

- **Avoid reinventing the wheel**. It is important to pull on lessons learned from other disciplines related to methods and uptake. In AQUAS we had some scouting exercises into other domains by mostly non-specialists (of those domains), the purpose being to consider mutual benefits. There appeared synergies between Technical Debt and DCE related to traceability and potentially also useful for triggering AQUAS IPs. DCE was seen to be an enabler for Agile Engineering at system level, for concurrent engineering and also for AI and IoT advancements.

7 Further Identified Challenges

Although AQUAS has achieved its goals as stated in the proposal, there are still challenges to be overcome in the future, some in relation to the extensive scope of dependability co-engineering and also partly because their implementation exceeds the project runtime. This chapter presents therefore the open challenges that need to be resolved in the time after AQUAS will have been completed.

7.1 Integration of the human factor in dependability co-engineering

The discipline of human factors has for safety-critical systems a very similar role to that of security. The primary need for applying both disciplines to these systems is that without their contribution, the top requirement that these systems be safe enough cannot be achieved. Multiple industrial sectors have given ample demonstration of this, with e.g. cars that can be remotely highjacked by criminals to cause accidents, uranium enrichment centrifuges that can be ruined by malware, and aircrafts that crash because of wrong assumptions on the opportunity they afford pilots to cope with the effects of quite common sensor failures. Of course, both disciplines may also be needed to satisfy other requirements that are specific to security or to usability alone, like protecting from unauthorised eyes the confidentiality of proprietary software in a system in operation, or of patient data that are fed to an operating theatre equipment; or ensuring high productivity of the human users of such equipment.

But in any case, the challenge of co-engineering these various qualities has at least two aspects:

- the analysis methods routinely applied to one discipline, with the modelling languages and the tool supporting them, were not invented to interface well with those employed in another, even when they simply use different names and symbols for similar concepts
- the cultures and languages of the specialists do not communicate well. This reinforces barriers against co-engineering, and the underestimation by specialist of the importance for their design goals of factors outside their area of direct expertise.

Yet to analyse risks in a design concept or implementation, and methods for countering them, it is essential at some stage to analyse how the phenomena that are familiar to the experts of the various disciplines affect one another, and to represent them together in the analysis (both deterministic analyses of design and probabilistic analyses of their dependability) by which designs are compared and assessed to limit risk.

For instance, the small case study in AQUAS addressed the concern of protecting the safety of a patient against the risk posed by a malevolent individual entering commands in a life-critical medical device (a close-loop controlled infusion pump). Official guidelines may recommend that access to such devices is thus protected by some kind of user authentication. But if the authentication procedure is cumbersome and/or failure-prone (e.g. passwords, or fingerprint recognition), it may cause excessive safety risk (and efficiency loss for the user, which also affects, indirectly, safety). A further complication is that the users may well recognise these risks and reduce them by circumventing the security features (e.g. they might write the passwords on the device), so that the designer's attempt to improve security against a specific threat backfires, being at best useless and at worst opening more security vulnerabilities (e.g. if the organisation has a "single-login" policy), with possibly further detriment to safety. Analysing such interaction thus requires the ability of experts of the various disciplines to spot potential risks (e.g. the likely reactions of medical staff to an ill-designed security feature), plus an ability to analyse the whole web of cause-effect relationships and to some extent quantify the overall results, to be able to assess trade-offs and the effects of the various factors involved on the overall results in terms of safety, productivity or other.

The discipline of human factors (and especially its sector dealing with computers) is nowadays for most of its practitioners close to the "softer" social sciences: its experts are well practiced in identifying risks and solutions by expert analysis of systems and by empirical studies, both qualitative and statistical, on people interacting with machines; their studies mostly eschew quantitative predictions. Integration of human error probabilities into safety studies, while still practiced, has gained a bad reputation (possibly due to the legacy of over-simplistic methods in e.g. nuclear risk assessment). This style of analysis is difficult to merge in engineering analyses of the non-human elements of a socio-technical system.

Directions for progress include at least:

- application of rigorous verification tools to human factors. For instance, typical safety problems like mode confusion or the potential for a complex sequence of operator actions to lead to a specific hazard state can be tackled by model checking (e.g. [8]).
- integration of human behaviour in probabilistic modelling, including the variability of human performance and the effects of human adaptation (e.g. the above rebellion against security measures, or how operators adapt to high frequency of false alarms to the point of not responding to correct ones).

In both directions, there are examples of application at least for simple systems. The incidents mentioned above demonstrate how (i) tools and techniques for applying them well are not yet standard tools in everyday engineering; (ii) the cultural divide between the relevant disciplines is still excessive. Promising directions seem to be to integrate or extend modelling languages to serve as a proper common language between these disciplines and experiment more extensively with both formal verification and probabilistic modelling to include these various risk factors together. Research must include application of existent methods, as in the bullet points above, as well as of extended and new methods of co-engineering analysis to substantial case studies, to validate methods, provide motivation and examples for industrial adoption, support through their reporting cross-education between specialist cultures, and support evolution of standards.

7.2 More widespread offerings for training

Today there are several Safety training offerings at universities, this role is often also with certification-oriented enterprises like e.g. TÜV in Germany. However, they think in a mainly safety-oriented manner, and security training is usually separated from the former. So there are still too few that teach Co-engineering, even a recently advertised academic course for critical systems development seem to ignore the need for co-engineering.

There are several positive exceptions like AQUAS partner City University of London who are planning dedicated courses, and a couple of other universities in UK and the US that provide lectures on combined safety and security. Co-engineering guidance by standards is also in a very incipient state. So, the current generation of specialists is mostly perpetuating the isolated thinking in silos. There are conference and workshop contributions on co-engineering and journal articles may raise awareness. But the key lever to change the situation is raising a new generation of multi-concern aware engineers. Approaching the young generation is of paramount importance to overcome the separation, and co-engineering education & training are badly needed to sustainably break down the walls between the silos in the minds of the experts. Future projects in the co-engineering area should, thus, have a strong emphasis on training.

7.3 Risk of prioritising automation over dependability

A direction for the development of dependability co-engineering as a part of industrial practice is certainly the improvement of tools that mechanise the tedious, error prone or mathematically complex parts of analysis of designs and of their comparisons. However, a major difficult part of

controlling risk is acknowledged to be the expert task of identifying *what* needs to be analysed by these mechanical aids; or recognising whether the model being analysed (no matter whether by a probabilistic model, a model-checkers, a deterministic fault tree tool or WCET analyser) *actually represents* the aspects of the system that *ought to* be analysed so that potential flaws are identified.

As an example, hazard analysis is recognised as an activity that is very hard to perform well, and that requires essential human input – inventiveness, expertise and imagination. So, a human role that is very difficult to automate remains essential, despite the fact that many forms of automated aid can be adopted to facilitate it.

It is thus necessary to recognise not only the limits of automation but the ways it can *increase* risk, and take them into account both in pursuing improved automation and in identifying the right combination of human and automated activities, the right boundaries and interfaces between the two, which must take into account both the state of the art in automation of these activities and the culture in which automation is used.

Risks arise because (i) modelling tools that simplify analysis and hide its details from users lend themselves to being used inexpertly, so as to produce superficially convincing but dangerously simplistic (or simply wrong) analyses, with spurious demonstrations that a system is good enough; (ii) even competent users may end up (through adaptations in their habits or mental processes) unwittingly over-trusting in practice a tool that is generally useful, so as to make it occasionally dangerous. Research is needed to better define the essentially human preserves of activity, though possibly also exploring how to use software to aid human creativity and expertise, in identifying e.g. system hazards, or areas of design analysis that are especially error prone or need more intensive empirical verification; and also to better study how people in fact use and rely on automated design and analysis methods. Methods and tools should be engineered to reduce the risk of user de-skilling, so that engineers can retain enough critical faculties and understanding of the process to mitigate or contain the potential downsides above; and to develop functions that assist users in the actual part(s) of the process where they need the most help. A possible direction might be that of improving the explanatory abilities of the tools so that they can have a training function.

7.4 Standards evolutions towards standardised co-engineering

Standardisation is of special importance in supporting dependability co-engineering, and for future challenges in CPSoS development.

There are still gaps that have not yet been answered and standards not yet addressed. Standardisation is a dynamic and evolving environment that has to be influenced actively, it requires windows of opportunities and time, and even obtaining awareness on the needs for co-engineering will span beyond the project limit to achieve some alignment with the project outcomes.

For example, the recent new requirement in the IEC 61508-3 and IEC 61508-1/2 for the inclusion of cybersecurity in the Risk and Hazard Analysis took a lot of effort and discussions among members to accept cooperative work between Safety and Security teams that last two years.

7.5 Enhancing policies in Europe/worldwide

As mentioned above in section 7.2, education and training are an important vehicle for sustainably leveraging the penetration of industry with the novel methodology. However, the challenges still to be overcome, as described in this document, require a sustainable basis to intensify the usage of the novel methodology and also advance it in terms of higher automation and adaptation to today's

trends in industry like IoT, Autonomous Driving, Artificial Intelligence and Machine Learning, Agile or Concurrent development.

In order to sustainably cope with the future technical challenges and implement ubiquitous dependability co-engineering in critical systems industry, guidance must come from regulatory bodies, and national, European and international legislation must enable conditions under which research policies and programs explicitly encourage and fund DCE research. They must demand the simultaneous consideration of the different dependability aspects throughout the entire product lifecycle requiring also standardization bodies to link functional safety and cyber-security standardization accordingly, which is currently only showing first attempts in the automotive industry by linking safety in ISO 26262 with cyber-security in ISO/SAE 21434. Also, certification must be adapted for the co-consideration of safety and security. Special incentives should support the transition for SMEs, encouraging the formation of start-ups with respective specialist knowledge.

At the end of the day we need a specialized dependability roadmap that goes beyond what was discussed in the CPSoS and Platform4CPS roadmaps, briefly covered at the beginning of this report, and a strategic master plan as well as research funding programs which explicitly target co-engineering at least as a prominent horizontal topic, if not the entire program is dedicated to DCE. Such a dedicated program could be a new joint undertaking involving funding from the European industry together with the EU research budget. High visibility of the DCE theme could also be achieved by founding a non-profit organisation which bundles the energy of volunteering enthusiasts building a network to spread awareness and ideally also develop solutions.

7.6 Quantifying the benefits of DCE

Despite the soundness of DCE approaches, it still quite a challenge to quantify the benefits of their integration in existing product lifecycles of companies. In addition, the adoption of practices depends on several factors. For instance, for DCE a co-engineering culture should be initiated and established to remove acceptance barriers. Also, it will be required to measure the risks and business justifications for the transition to higher levels of DCE. In AQUAS we have analysed the improvements on automation compared to previous mostly manual DCE practices. We have also analysed the synergies between standards related to safety and security to avoid redundant work. Finally, we have applied cost models, like the Error Management Compass, to provide an economic perspective for specific combined analyses that were experimented during AQUAS. In this cost model, the impact of both cost and product quality are analysed in the context of specific use cases. The open challenge is to provide more evidences on business justifications of DCE. Currently, our results cannot be generalised for all companies and we do not claim that a full generalization is possible. Thus, cost models can be enhanced with ways to gather accurate estimations for a better analysis of the potential benefits of DCE adoption.

8 Summary and Conclusions

The AQUAS project has proposed an Interaction Point-based approach for dependability co-engineering with the goal of improving the quality and reducing cost, risk and time for developing, operating, maintaining, and decommissioning complex critical systems. The methodological basis has been analysed and described, and tool features targeted towards co-engineering and combinations of tools enhanced for AQUAS, so-called Prototypes, have been developed. Finally, case studies from five exemplary domains implemented demonstrators to validate the approach.

In this deliverable we have described the achievements in AQUAS with respect to co-engineering functionalities and the challenges that need to be overcome in the future. Both aspects were analysed from the methodological perspective, from the viewpoint of the five use cases, and from the tools point of view.

A number of issues that should be improved in the future were detected during project work. As an example, in the case studies, the shift of effort from test and verification to earlier phases could be clearly observed as a positive result. On the other hand, potential for smoother tool interoperability was detected, in particular when the tool collaboration extends across different PLC phases. This potential for improvement should be addressed in the future.

Unresolved challenges that were detected during the project are only partly of technical nature, so, for instance, the lack of guidance by appropriate standardisation was identified as an open issue. And there are many more: A higher degree of automation in co-engineering, dedicated education and training for co-engineering, motivation for industry and incentives for SMEs to adopt the co-engineering approach are only a few examples. Another future challenge is to address more domains, for instance Autonomous Driving.

Summarizing, we may state that a research programme to advance the achievements further and address mode domains, the development of dedicated combined standards, and academic courses on co-engineering should be the key levers to maximize the benefits of AQUAS in the future.

9 References

[1] D1.9 - Report on the Evolution of Co-Engineering Standards, AQUAS project, http://aquas-project.eu, 31.10.2019.

[2] CPSoS Roadmap – result of project CPSoS – Towards a European Roadmap on Research and Innovation in Engineering and Management of Cyber-physical Systems of Systems https://www.cpsos.eu/

[3] Slijvo, I., Gallina, B.: Building Multiple-Viewpoint Assurance Cases Using Assumption/Guarantee Contracts. 1st International workshop on Interplay of Security, Safety and System/Software Architecture (ISSA-2016), 39. DOI, Open access version

[4] Javed, M. A., Gallina, B.: Towards Variant Management and Change Impact Analysis in Safety-oriented Process-Product Lines. 34th ACM/SIGAPP Symposium on Applied Computing (SAC 2019). DOI, Open Access Version

[5] Gallina, B.: Quantitative Evaluation of Tailoring within SPICE-compliant Security-informed Safety-oriented Process Lines. Journal of Software: Evolution and Process, Vol. 32, Issue 3, August 2019

[6] K. Taguchi, et al. *Threat Analysis Framework for Safety Architectures in SCDL*. in *SAFECOMP*. 2020, (accepted for publication). Springer.

[7] D3.2 - Combined Safety, Security and Performance Analysis and Assessment Techniques – Preliminary, AQUAS project, http://aquas-project.eu, 10-05.2019.

[8] John Rushby, "Using model checking to help discover mode confusions and other automation surprises"Reliability Engineering & System Safety, Volume 75, Issue 2, February 2002, Pages 167-177. https://doi.org/10.1016/S0951-8320(01)00092-8

[9] Recommendations for Security and Safety Co-engineering - Part A, S. Paul et al., Thales Research & Technology, MERgE Project, Feb. 2016.

[10] C. Schmittner, T. Gruber, P. Puschner, E. Schoitsch: "Security Application of Failure Mode and Effect Analysis (FMEA)", Safecomp 2014, Florence, Italy, proceedings, pp 310-323.

10 Abbreviations

AMESim systems	a commercial simulation software for modelling and analysis of multi-domain
ASIL	Automotive Safety Integrity Level
ATM	Air Traffic Management
CE	Co-Engineering
CHESS	an Eclipse-based model-driven engineering framework based on Papyrus
CPSoS	Cyber-Physical System of System
CPU	Central Processing Unit
DAL	Design Assurance Level
DCE	Dependability Co-Engineering
EA	Enterprise Architect (modelling tool by Sparx Systems
Eclipse	an Open <source modelling and programming environment www.eclipse.org
ECSEL	Electronic Components and Systems for European Leadership
FMEA	Failure Modes and Effects Analysis
FMECA	Failure Modes, Effects and Criticality Analysis
ICT	Information and Communication Technologies
IoT	Internet of Things
IP	Interaction Point
MARTE	(a UML 2 profile for) Modelling and Analysis of Real-Time and Embedded systems
ML	Machine Learning
MoMuT	model mutation-based test case generation, a tool suite of partner AIT
OpenCert	Eclipse-based multiconcern-assurance platform from projects SESAMO and AMASS
Papyrus	An Eclipse based modelling environment
PLC	Product Lifecycle
ProVerif	an automatic cryptographic protocol verifier
QEMU	“Quick Emulator” – a generic and open source machine emulator
RIA	Research and Innovation Action
SAN	Stochastic Activity Network
SIL	Safety Integrity Level
SSP	Safety, Security, and Performance
SW	Software
SysML	Systems Modelling Language (a standardized UML 2 based modelling language)
SystemC	Modelling&simulation language for complex electronic HW/SW component systems

ThreatGet	EA plugin for rule-based multi-concern analsysis of models with dependability properties
TOF CUff	a neuromuscular transmission monitoring system by RGB
Ttool	toolkit with UML/SysML diagram editor and simulation/formal verification of SSP
UC	Use Case
UML	Unified Modeling Language
UML 2	UML version 2
WCET	Worst-Case Execution Time

11 Glossary

For brevity we include here some of the more specific reference list we used in AQUAS (quite a longer list existed especially with clarifying terms for safety and security). These were either defined through discussions with the consortium or referenced/derived from these sources: ECSS, SEBoK, A. Avizienis, J.-C. Laprie, Brian Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," IEEE Transactions on Dependable and Secure Computing, FIPS 200, ISO/IEC 15288:2015 - [SOURCE: ISO Guide 73:2009, ISO/IEC/IEEE 15288.

Term	Definition
AQUAS Approach	Supports the evolution of industry towards applying the AQUAS Methodology. It is a progressive collaboration framework for both short and longer-term needs that advances both the tools and the product process (for co-engineering). It considers both technical and non-technical needs including evolution of standards to be more adapted towards dependability co-engineering.
AQUAS Methodology	Focused on the optimisation and automation for tradeoffs related to the coupling between safety, security, performance and usability. This is being referred to as Dependability Co-engineering. It includes: • Analysis of the needs of industrial application domains. • Giving support for balancing existing safety & security requirements with application specific performance requirements • Established tools and platforms, which will be upgraded to implement and test the co-engineering approaches and improved processes and methods • Taking into account the complete product lifecycle. • The capability for reliable system integration when sourced from many subcontractors and capability for systems to evolve such as when some hardware is replaced or there are software upgrades.
Dependability Co-engineering	(AQUAS) Currently representing the coupling between Safety, Security, Performance.
Technology	Used to encompass both methodology and tooling.
PLC phases	Project specific terminology - 'phases' has been used to provide a degree of fuzziness on the separation of the lifecycle. This is because the use cases have differing PLCs but also because the architecting part was split into modelling and simulation. 'PLC stages' is the standard nomenclature.
Modelling Phase of PLC	Project specific terminology - This is encompassed by the architecting stage of the PLC, representing in particular the methods/tooling producing static architecture representations (i.e. w/o timing or behaviours).
Simulation Phase of PLC	Project specific terminology - This is encompassed by the architecting stage of the PLC, representing in particular the methods/tooling producing dynamic architecture representations (i.e. with timing & behaviours).
Co-engineering	This represents managing the interactions between different engineering focuses. In AQUAS these are the system qualities safety, security, performance and also usability. In particular it orchestrates the manual and automatic trade-offs within and across stages of the product lifecycle.

Interaction Point (IP)	(AQUAS) This is considered both an activity to resolve trade-offs and the point in a product life cycle (PLC) at which it occurs. The activity is "interaction" in that (a) experts in the various aspects of the system and its properties interact., e.g. security and safety experts; (b) providing an IP combined analysis of particular interdependencies (c) the need for changes or decisions may be recognised that require an integrated view, e.g. because of inevitable trade-offs between desirable properties, and these trade-offs are discussed between the various experts to produce recommendations/decisions.
IP Combined Analysis	(AQUAS) Where the coupling between separate system analyses such as performance and security are considered together, that may be anywhere in the range from informal discussion and mutual critique to using mathematical models to assess various measures of interest for alternative design options, or even a single, summary measure to be optimised (e.g., probability of an undesired event). The analysis takes place through a mix of automated and human effort and may require several iterations. Note IP should precede the term combined analysis to avoid misinterpretation that we are fusing the safety and security analyses (concerns are separated to manage complexity).
Trade-off Artefact	(AQUAS) This is the representation of a decision balancing an interdependency between several system properties (e.g. security & performance). It is stored in a database and represents the relation between the properties and the specific levels chosen. It includes the acceptable ranges in which these properties can evolve when one of them is changed is required. They are connected within and across phases of the product lifecycle. This means it is possible to alert the stakeholders when system modifications (such as a system patch) require system property ranges to be reassessed. The automated traceability is likely to only relate to high-level criticalities due to cost constraints. It is possible a trade-off artefact might also include the decision rationale, history and other decision support aids. E.g. Steps for a particular phase: [Interdependencies Identified] -> [Criticality Levels Identified] -> [Trade-offs agreed] -> [Added to Artefacts Database] E.g. Steps where a previous decision is impacted: [Analyse Related Artefacts] -> [Potentially Update Artefacts Database] -> [Then Alert Stakeholders Across PLC]
Product Lifecycle	Evolution of a system, product, service, project or other human-made entity from conception through retirement.
PLC phases	Project specific terminology - 'phases' has been used to provide a degree of fuzziness on the separation of the lifecycle. This is because the use cases have differing PLCs but also because the architecting part was split into modelling and simulation. 'PLC stages' is the standard nomenclature to use in publications.
Modelling Phase of PLC	This is encompassed by the architecting stage of the PLC, representing in particular the methods/tooling producing static architecture representations (i.e. w/o timing or behaviours). 'Stage' is preferred usage outside AQUAS.
Simulation Phase of PLC	This is encompassed by the architecting stage of the PLC, representing in particular the methods/tooling producing dynamic architecture representations (i.e. with timing & behaviours). 'Stage' is preferred usage outside AQUAS.
Traceability	For AQUAS this particularly applies to visibility of interdependencies of qualities across the PLC phases. E.g. there may be a change in security functionality in the operational stage which needs a tradeoff adjustment of safety functionality in the architecting stage.

Safety	State where an acceptable level of risk is not exceeded. This may apply to the system or its environment (particular to people).
Safety Engineering	As an engineering discipline, system safety is concerned with minimizing hazards that can result in a mishap with an expected severity and with a predicted probability.
Performance	Quantifiable characteristics of a function.
Security	State where an acceptable level of risk arising from malevolent action is not exceeded.
Security Engineering	Security engineering is concerned with building systems that remain secure despite malice or error. It focuses on the tools, processes, and methods needed to design and implement complete systems that proactively and reactively mitigate vulnerabilities. Security engineering is a primary discipline used to achieve system assurance.
Dependability	Dependability of a computing system is the ability to deliver service that can justifiably be trusted
System Qualities or Attributes	These are properties at the system level that affect many functions within the system. Often referred to as extra- or non-functional properties. Represented by SSP in AQUAS.
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 system given the potential impact of a threat and the likelihood of that threat occurring.
Risk (2)	Effect of uncertainty on objectives Note 1 to entry: An effect is a deviation from the expected — positive or negative. A positive effect is also known as an opportunity. Note 2 to entry: Objectives can have different aspects (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product and process). Note 3 to entry: Risk is often characterized by reference to potential events and consequences, or a combination of these. Note 4 to entry: Risk is often expressed in terms of a combination of the consequences of an event (including changes in circumstances) and the associated likelihood of occurrence. Note 5 to entry: Uncertainty is the state, even partial, of deficiency of information related to understanding or knowledge of an event, its consequence, or likelihood
Threat	Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, or individuals through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service. Also, the potential for a threat-source to successfully exploit a particular information system vulnerability.
Fault	A fault is the adjudged or hypothesized cause of an error. A fault is active when it produces an error, otherwise it is dormant.

Trustworthiness	Worthy of being trusted to fulfil whatever critical requirements may be needed for a particular component, subsystem, system, network, application, mission, enterprise, or other entity

12 Appendix 1: AQUAS Combined Analysis Methods

The methods of combined analysis used in the AQUAS project are listed in Table 2. Each method can be used in one or more phases of a typical PLC, which are also shown in Table 2.

Table 2. AQUAS methods of combined analysis applied to PLC phases

		PLC phase			
	Combined analysis description	Requirement/ concept	Detailed design	Verifying implementation	Summary of the method
1	HazOp for safety/security interactions	✓			
2	Interference analysis (using medini analyze)	✓			medini analyze produces an SSP interference table based on functional and non-functional requirements and their allocation to components
3	Combined Hazard Analysis and Threat Assessment (with fault/attacks trees)	✓			
4	Combined Safety, Safety & Performance Analysis	✓		✓	the Enterprise Architect plugin ThreatGet allows modelling a system, defining safety security and performance rules, analysing their fulfilment and managing risks to the system.
5	Security and Performance analysis	✓			
6	Safety & Performance Design Space Exploration		✓		
7	Security and Performance (using SSDLC and TTool)		✓		SSDLC selects the right security mechanisms and their complexity which is fed up in TTool model in order to study the impact of these mechanisms on system performance
8	Interference analysis from a safety-security combined analysis and performance analysis (Concept-		✓		The safety-security combined analysis generates combined trees: including safety and security elements. The interference analysis provides

	aware analysis tool, Safety Architect, Cyber Architect, CHESS).				high-level reports on the interdependence of safety and security using assets from the combined analysis. A schedulability analysis is applied early in the design phase to check and demonstrate that, under the WCET assumptions, the performance requirements are feasible.
9	Trade-Offs Regarding User Authentication	✓	✓		
10	Performance-Security Trade-Offs via SANs	✓	✓ with measure ments		A state-based probabilistic model (Stochastic Activity Networks) is used in which performance is modelled as a function of the state of a load model (e.g. light load, medium or high load). This model allows one to compute the probability that a message latency exceeds a defined threshold.
11	Safety-Security-Performance Trade-Offs via SANs	✓	✓		A state-based probabilistic model (Stochastic Activity Networks) in which the states of different components are modelled as state-machines, the transitions between the states are driven by events – errors/failures or successful attacks and repairs/cleansing of software. The model allows one to compute the probabilities that a mission of given length ends with unsafe or safe failure or false alarm.
12	Performance implications of meeting security requirement (using TTool)	✓	✓		
13	C Code Conformity			✓	
14	Verification of System SW Using Ada, SPARK			✓	
15	Model-Based Testing for Multiple Concerns (with MoMuT::UML)			✓	Random and coverage driven generation of test sequences from behaviour models extended by ML (Machine Learning) of the

					performance predictor to evaluate the expected performance during exploration of the state space and create a set of performance and also safety test cases.
16	Environment simulation (SystemC + TTool)			✓	
17	Multiprocessor Task Scheduling		✓	✓	Tasks are assigned to processors according to joint safety/security/performance optimisation criteria.

13 Appendix 2: Tools and Tool Combinations

The following table contains, after the list of tools, also the list of combinations of tools realizing an AQUAS CE functionality

Table 3 AQUAS dependability co-engineering tools and tool combinations

Partner	Tool / Tool chain	Dependability co-engineering functionality	Challenges/future plans after AQUAS
Individual tools			
Astrée/ RuleChecker	RC / A-RC / Asserts	Derives unprecedented levels of confidence in safety and security of code as input for trade-off analysis. Astrée's full-semantic analysis of SYSGO's PikeOS proves the absence of runtime errors, coding guideline violations, and other safety- or security-related code defects. Adaptions made to the interpretation of MISRA rules allowed SYSGO to safely discard false and unintended alarms that, in the past, required manual inspection and justifications.	All adaptions to Astrée/RuleChecker will be integrated in its commercial version. The work done with Astrée in AQUAS serves as a proof-of-concept for sound semantic analysis of an OS. The adapted coding guideline checks meeting SYSGO's interpretation of MISRA rules ought to be relevant for other OS providers and development teams in general. The achievements made in AQUAS will be used to increase market visibility and attract new customers from OS development.
TimingProfiler	WCET estimation	Derives performance results (estimated worst-case execution times of non-interrupted code snippets) that can be used directly as input to a trade-off analysis, or can be communicated to the A2K tool by a dedicated tool interface. The A2K tool uses this input when it explores the design space of concurrent code safety and system timing performance.	All adaptions to TimingProfiler will be integrated in its commercial version. The improved tool will be offered to new and existing customers. The adaptation of TimingProfiler to the space-multicore use case and its interfacing with ITI's A2K increase its capabilities and its applicability for the analysis of software running on multicore processors.
TrustPort	SSDLC Security Validation	During AQUAS project there were added new security requirement catalogues and implemented new functionality for validation and verification in whole PLC concept.	Future product development will be focused on specific application areas (smart grids, smart metering, IoT). The new verification methods were planned to be implemented and verified in real scenarios/pilots project. SSDLC will be used in future design, implementation and
	SSDLC security/performance validation		

			verification of security requirements with performance trade-offs (e.g. in cybersecurity for IoT - bottle neck of security of low performance LPWAN/IoT or security/performance trade-offs in smart metering rollouts).
A2K (See ITI prototypes below)	Timing & Safety Analysis		
	Absint WCET Estimation		
	BUT Tools OSLC		
Concept-aware software assets analysis prototype	Core	The focus of the Concept-aware tool is to provide interference analysis capabilities on safety and security using assets from different stages of the product life-cycle. Interference analysis refers to techniques analysing the mutual influence and inter-links of different quality attributes. Our interference analysis provides high-level reports on the interdependence of safety and security to reveal and trigger the need of a co-engineering meeting and to visualise and monitor the evolution of the safety and security interdependence. Previous interference analyses were mostly manual, the library and tool was created to fill this gap and to be easily integrated with other tools.	The Concept-aware software assets analysis tool and library developed in AQUAS to support interference analysis will serve to extend the capabilities of Eclipse OpenCert and the Sabotage safety assurance framework, thus it can be part of the offer of services around them, and around other potential tools given its integration and extensibility capabilities.
	Analysis of the evolution		
	Fault trees support (and integration with ALL4TEC)		
	Requirements support		
	System design support		
	Assurance cases support		
AMT	Integration of DOORS NG	The implementation of this connector is essential to integrate medini analyze in the tool landscape of customers and allows the bi-directional interchange of requirements.	All adaptions to medini analyze will be integrated in its commercial version. The improved tool will be offered to new and existing customers.
	Asset Identification	The asset identification supports the annotation of cybersecurity related attributes to SysML models in medini analyze	

	Threat Identification based on STRIDE	Based on the attribution of the SysML models potential threats for later assessment are derived by applying the STRIDE threat categories	
	Threat Assessment	The threat assessment helps to understand the impact and the feasibility of the potential threats. The result is a risk level that allows for risk-based system design	
	Threat Treatment	According to the risk level the threat treatment supports to definition of strategies to deal with the identified risk.	
	Attack Trees	The attack trees in medini analyze support the description of attack scenarios leading to the potential threats that were derived from the attributed SysML model.	
Intecs Solutions	medini-CHESS integration	medini – CHESS integration was improved to import entities from a source medini model into the target CHESS model and extended to create traceability links between medini and CHESS model entities. Traceability is useful for navigation purposes and to maintain the synchronization between the two models.	
	Traceability support	The CHESS Traceability Feature prototype provide support for the visualization and management of traceability links between requirements and design entities.	
	CHESS Support for co-analysis via SAN	We have extended the CHESS dependability profile and modelling capabilities to support safety and security co-engineering, and automatic transformations to SAN model for reliability analysis with MOBIUS. This approach provides a smooth integration, guarantees the consistency among SysML and SAN models, and drastically reduce the effort required to construct an appropriate SAN analysis model.	The extensions developed in AQUAS are delivered under Eclipse as open source, and will be enclosed in a new CHESS major release, and will contribute to improve the CHESS open methodology and the support tool offered.
ALL4TEC	Import from CHESS to Safety Architect	The Chess import function has been improved to get a more comprehensible imported model in Safety Architect.	Further work addressing safety-security analyses will be carried out, in particular by considering a new security risk analysis method: EBIOS Risk Manager and thus establish a link with a tool dedicated to this method:
	Fault trees export in OpenPSA	The fault trees import function in OpenPSA format has been enhanced to automatically add tags on the nodes of	

		the trees and to be compatible with the concept-aware analysis tool from Tecnalia.	Agile Risk Manager.
	Combined Safety-Security local analysis	Diagram viewpoint allowing to describe relation between safety and security elements.	
	Attack Trees in Cyber Architect	Implementation of the attack tree function in Cyber Architect. The diagram representation (attack tree) of attack scenarios is helpful for the representation of safety-security analysis also based on diagram.	
CEA	Frama-C - Analysis of C code generated from B	Provides feedback that architecture post trade-offtradeoff analysis is implementable in the software.	Maturation of the tool to support more properties derived from B.
	B0 to ACSL translator	Necessary to ensure that what is analysed in the code is conform to design requirements derived from requirements trade-offtradeoff analysis. Requirements in B0 format.	Maturation of the tool to support more constructs expressible in B0.
	Papyrus Software Designer - C generator	Bridge between design requirements, derived from trade-offtradeoff analysis, and implementation.	Support for component-based models and distributed systems with middlewares.
	Papyrus Software Designer - ACSL generator	Necessary to ensure that what is analysed in the code is conform to design requirements derived from requirements trade-offtradeoff analysis. Requirements in UML format.	Automate ACSL specifications from design requirements.
	ACSL Xtext editor	Specify implementation-level requirements from design requirements post trade-offtradeoff-analysis.	Support more constructs of ACSL.
AIT	GSFlow Workflow engine with requirements management	In AQUAS, the workflow engine GSFlow was extended to support the integration of security requirements created by TrustPort's SSDLC tool.	-
	ThreatGet tool for automated rule-based SSP analysis	The rule-based security analysis tool ThreaGet was extended by safety and performance analysis features in AQUAS . In particular, the set of model elements and the rule grammar were adapted to satisfy the needs of the	The extended features shall be made mature and commercially available via industry partner LieberLieber. Future research is intended to extend the analysis engine for supporting the creation of combined fault-attack

		industrial drive use case architecture and to extend the covered properties from security to safety and performance.	trees, considering here also threat and failure propagation.
	MoMuT model based test case generation	Random and coverage driven generation of test sequences from behaviour models provided by MoMuT was extended by ML (Machine Learning) of the performance predictor to evaluate the expected performance during exploration of the state space and create a set of performance and also safety test cases.	The experimental features added in AQUAS shall be migrated into publicly available releases earliest in autumn 2020. Using the performance predictions for a heuristic search over (sub-sets of) the state space might be implemented at a later stage. Currently the ML based task needs human experience and know-how to come up with a good prediction model. A higher degree of automation in selecting the prediction model is envisaged for future research projects.
MTTP	TTool	• Addition of an automated performance analysis engine so as to study the impact on performance when adding safety/security mechanisms • Rework of the internal model-checker in order to support addition safety properties (e.g. in CTL) • Addition of input / output capabilities in order to better interact with other tools (e.g. SSDLC, Amesim)	• Explicit integration of Interaction Points • Automated assistant for proposing mechanisms that could help resolve verification failures while keeping other properties verified
Tool combinations			
ITI Prototype 1	Multicore System Safety & Performance Analysis with A2K	A2K enables design space exploration of concurrent code safety and analysis of system timing performance. We developed a traceability of analysis feature during the AQUAS project, interconnected our analysis tool with BUT code analysis tools using OSLC, and interfaced with AbsInt timing analysis tools as well.	To extend the timing analysis algorithms to enable analysis of much more complex systems whose components have individual and independent scheduling policies.
ITI Prototype 2	Verification Environment for Medical Devices	The medical device evaluation environment allows co-engineering analysis and verification of the operation of the device under a wide variety of operating conditions.	To extend the medical device simulation and testing environment to include other types of drugs and patient models. Right now, we are working on including a

			hardware-in-the-loop environment for control of muscle relaxation anaesthesia.
Tecnalia Prototype	Concept-aware software assets analysis	See previous row on the “Concept-aware software assets analysis prototype”	See previous row on the “Concept-aware software assets analysis prototype”
TRT Prototype	GR712 Applications performance characterization	Contributes to benchmarking performance changes with respect to variations of safety or security properties of a system	Advancing on coupling with safety and security for system variations to support design & automation. This supports also transfer capability (adapted for conditions/environment of use).
TrustPort Prototype	Software Development Life Cycle Management Tool (SSDLC)	According to up to date practise in cybersecurity domain, the verification methods, tools and set-up for evaluation proper design and implementation are in demand (e.g. example in SSDLC for particular requirements). Outside security/performance/safety co-engineering, another important relationship was revealed, it is between security and usability. Security policies can be self-defeating if they reduce usability of the security mechanisms or of the systems they protect: for example, requiring complex passwords to improve security may cause users to respond by sharing passwords, having one password for many devices, keeping passwords written down next to the protected devices, reusing or recycling old passwords, etc.	Future product development will be focused on specific application areas (smart grids, smart metering, IoT). The new verification methods were planned to be implemented and verified in real scenarios/pilots project. SSDLC will be used in future design, implementation and verification of security requirements with performance trade-offs (e.g. in cybersecurity for IoT - bottle neck of security of low performance LPWAN/IoT or security/performance trade-offs in smart metering rollouts).

14 Appendix 3: AQUAS Partners

Table 4 AQUAS partners

Short name	Partner organisation name	Country	Contact	Topics covered in AQUAS
TRT	THALES Research & Technology	France	https://www.thalesgroup.com/en/global/innovation/research-and-technology	Proposal Coordination, Exploitation leader
TASE	THALES Alenia Space Spain	Spain	https://www.thalesgroup.com/en/global/activities/space	Project Coordination, space multicore use case leader
ISYS	Integrasys SA	Spain	https://www.integrasys-space.com/	ATM Use case leader
RGB	RGB Medical Devices, SA	Spain	https://www.rgb-medical.com/	Medical use case leader, usability and standardization
City	City University London	UK	P.T.Popov@city.ac.uk, L.Strigini@city.ac.uk	Leader WP3 Methodology; Human aspects; method provider (probabilistic modelling)
AIT	AIT Austrian Institute of Technology GmbH	Austria	https://www.ait.ac.at/	CE goal leader, standardization contributor, tool & method provider.
UNIVAQ	Università degli Studi dell'Aquila	Italy	https://www.univaq.it/	Tool / method provider
SISW	Siemens Industry Software	France	www.siemens.com/plm	WP4 design / tooling leader
MDS	Magillem Design Services SA	France	http://www.magillem.com/	Tool provider
ClearSy	ClearSy	France	https://www.clearsy.com/	Railway use case leader
CEA	Commissariat à l'énergie atomique et aux énergies alternatives	France	http://www.cea.fr/	Tool / method provider
TrustPort	TrustPort, a.s.	Czech Republic	https://www.trustport.com/en	Tool provider (Security requirement management in PLC)

Short name	Partner organisation name	Country	Contact	Topics covered in AQUAS
MTTP	Institut Mines-Telecom, Telecom ParisTech	France	www.telecom-paris.fr	Tool / method provider
Tecnalia	Tecnalia	Spain	https://www.tecnalia.com/en/	PLC goal leader
BUT	Brno University of Technology	Czech Republic	https://www.vutbr.cz/en/	Dissemination leader
A4T	All4Tec	France	https://www.all4tec.com/	Tool provider
ITI	Instituto Tecnológicio de Informática	Spain	https://www.iti.es/	Tool provider (modelling and timing analysis of heterogeneous , multiprocessor systems)
Intecs	Intecs Solutions SpA	Italy	http://www.intecs-solutions.it/soemens	standardization evolution goal leader, method provider
SAG	Siemens AG Austria	Austria	https://new.siemens.com/at/de.html	Case study WP leader, Industrial Drive Use case leader
HSRM	RheinMain University of Applied Sciences	Germany	https://www.hs-rm.de/en/	Tool / method provider
AMT	Ansys medini Technologies AG	Germany	https://www.ansys.com/products/systems/ansys-medini-analyze	Tool provider / method provider
SYSGO	SYSGO GmbH	Germany	https://www.sysgo.com/	Tool provider (PikeOS microkernel and hypervisor, configuration tool))
AbsInt	AbsInt Angewandte Informatik GmbH	Germany	https://www.absint.com/	Tool provider (verified compiler; static program analysis for stack usage, worst-case execution time, and run-time errors)