Will safety-security co-engineering pay off?

A quality and cost perspective in two case studies

Imanol Urretavizcaya,

Jabier Martinez,

Giuseppe Satriani,

Alejandra Ruiz

*Tecnalia, Basque Research
and Technology Alliance (BRTA)*

Derio, Spain

name.surname@tecnalia.com

Ricardo Ruiz Nolasco,

Antonio González

RGB Medical Devices

Madrid, Spain

{rruiznolasco agonzalez}@rgb-medical.com

Isaac Moreno,

Abel Balbis

Thales Alenia Space

Madrid, Spain

name.surname@thalesaleniaspace.com

Abstract—Safety and Security concerns are usually interlinked while building critical software-intensive systems of systems. Several efforts try to approach both domains of expertise to increase the overall reliability of the systems and reduce costs by an earlier detection of issues and trade-offs. Despite the growing number of co-engineering practices at different life-cycle stages, there is a lack on business justifications such as economic costs of their adoption. We report on using a cost model to evaluate the convenience (or not) of adopting co-engineering practices in two industrial case studies (space and medical devices). Simulation results with the collected data suggest an improvement in quality if any of the selected co-engineering practices are integrated while cost increases in one case but reduces in the other. We discuss the results but, as they cannot be generalized, the main contribution is on proposing the cost model for answering the title's question.

I. INTRODUCTION

In the design and development of critical software-intensive systems, it is the responsibility of the safety engineers to reduce the risk of catastrophic consequences on the users and the environment until reaching acceptable thresholds [1]. Safety and security aspects are highly interlinked as usually security is needed to ensure safety [2]. However, the security engineering team is also a very specialized discipline trying to guarantee confidentiality, integrity, and availability [1]. In practice, safety experts and security experts within organizations are sometimes considered as “silos” with less interaction than it might be desired [3]. This can be explained because of this high specialization and the usage of dedicated methods and tools, as well as standards and regulations guiding their practices and product life-cycles. Also, a more general problem is on the governance of organizations such as a lack of effective work between parts and poor communication [4].

Some approaches tried to identify the similarities between established safety and security standards to avoid potential redundant work [5]. There is also an extensive literature and concrete examples on co-engineering approaches [6]–[8] trying to harmonize both worlds and providing methodological and technical support for interaction points [3]. The main goal of these approaches is the early detection of issues and

trade-offs belonging to the safety and security interlinks so they can act accordingly.

Despite the soundness and promising results of several of these approaches, business and economic justifications are harder to find. Economic considerations are usually an important criterion for the adoption of a practice, specially when the adoption might imply substantial changes in the way of working of the organization.

We investigated the quality and economic impact of integrating co-engineering practices in a life-cycle using a cost model in two use cases from two relevant software-intensive and critical industrial domains; the medical devices and the space domain. Cost models [4], [9], [10] aim to capture the information about a system and its context with the objective to provide an economic perspective. In our case we used part of the Error Management Compass (EMC) [4]. Our approach is novel in looking at the adoption of co-engineering practices from the economical and quality points of view by proposing a systematic approach.

Although the results cannot be generalized to other companies or types of projects, the findings of the cost model suggest that the integration of the safety-security co-engineering practices in their baseline life-cycles represents improvements in terms of quality. Regarding cost, in the space case study the cost increases which is the normally expected result when you add new activities in a life-cycle. However, in the medical use case, the cost is also reduced suggesting that the cost of integrating practices compensates the cost of fixing issues that were not detected by these practices but later. We contribute then two empirical analyses and the evidence that the EMC, previously used in non-critical systems [4], can be used in critical-systems to repeat this kind of economic assessments.

This paper is structured as follows: Section II and III present background information and related work to better understand and position this work. The case study design is explained in Section IV and the collected data is detailed in Section V. Then, the results are presented in Section VI and later discussed in Section VII including the threats to validity. Finally, Section VIII concludes and outlines future work.

II. BACKGROUND: THE ERROR MANAGEMENT COMPASS

The Error Management Compass (EMC) is a cost model based on quantitative and statistical process management to improve performance in software-intensive organizations. It is based on simulation from an annotated model of the life-cycle. As a clarification, the “Error” term is included in the EMC name but it does not refer to error in safety terminology where failure, failure mode, fault and error have precise meanings [11]. Here it has a more broader sense, equivalent to the concept of defect [10] or issue (e.g., missing requirement, incorrect design, source code bug, etc.). In this paper we will refer to issues and errors indifferently. Figure 1 illustrates, in a simplified way, the five sequential stages of the cost model:

Stage-1: Error Recording. Every error, incident, anomaly must be accurately and systematically registered to create a database with a set of data: Where the error has been injected, which detection activity has identified the error, and the actual cost for the error correction. Examples of detection activities (co-engineering practices in the specific case of this paper) will be presented in Sections IV-A and IV-B.

Stage-2: Error Analysis. Collected data must be accurately analyzed and corrective actions must be implemented to optimize both injection and detection activities.

Stage-3: Injection Table. Based on the collected historical data, the table shows how many errors are injected in the system in each engineering activity and how many errors are detected by the detection activities. This table allows to calculate the effectiveness of the detection activities.

Stage-4: Cost Model. Once the errors flow has been modeled in the injection table, based on the historical data related to the correction costs for each type of error, a cost model can be developed. This model allows to quantify in economic terms the cost saved if the organization is able to detect and correct errors as closer as possible to their injection phase.

Stage-5: Predictive Model. This model allows a project manager to take decisions about the right composition of detection activities trying to maximize the effectiveness of the detection activities while aiming to minimize the costs. The predictive model is based on a Monte Carlo simulation.

Among the relevant metrics that can be obtained from the model, there is an indicator to measure the effectiveness of the internal verification process: the EIC (Error Injection Coefficient); this indicator measures the probability that an error is not detected by the client in the system in operation. That means measuring the relationship between the errors found by customers during the validation and operation stages

and those detected internally before starting the stage where the system is in production. For instance, an EIC of 90% means that only 10% of errors (of all types) that can be introduced across the life-cycle escaped from all the filters.

III. RELATED WORK

Economic aspects in software engineering have been a traditional subject of study [9], [10]. In general, the software industry experiments high uncertainty in achieving project goals because of defects and errors. It is of interest to decide which measures need to be adopted in the life-cycle to avoid them without relying on intuitions. As illustrative example, decision makers can consider that increasing the budget to the testing team at the implementation level is the best option. However, a more in-depth analysis could have shown that a new practice to, for example, check, revise and complete requirements could be cheaper and obtain better results across the life-cycle.

We consider that models for safety-security co-engineering cost analysis require more research attention. In safety-security critical domains, safety comes first leaving economic considerations in the background, however, it is a of relevance for several stakeholders in systems of systems development (e.g., project managers), specially to know where and in which practices it is more optimal to invest.

A similar analysis to ours was the one by NIST [12] to measure the economic impact of an inadequate infrastructure for software testing. Using this report as source, an analysis on the aviation domain was presented [13]. They reported that the leakage of faults from the software components’ development stage drastically increased the rework cost in the aircraft industry. The cost model was mainly based on analysing the life-cycle and identifying where issues are introduced, where issues are found, and what was the estimated cost for issue removal. This way we consider our approach also aligned with this estimation paradigm.

In our previous works [4], the EMC was used on the data available in two IT companies; both of them providers of IT services in the outsourcing market. Their business was mainly based in processing customers’ “requests of work” while maximizing their own productivity. Requests were exclusively related to evolutionary maintenance (no major reliability needs) and in this work we are considering a safety-security critical domain. Thus, we were largely inspired by these kind of economic assessments in non-critical systems and their way to help taking informed decisions. In this work, we explored the usage of the EMC to deal with the peculiarities of safety-security critical systems and the involved co-engineering practices.

In this work we focus on quality and cost regarding changes in the life-cycle (e.g., introducing new practices) and not in direct evaluations of design alternatives of the product itself. As example of the latter, cost is estimated in [14] for software safety functions in electric vehicles. They estimate the use of a new technical reference architecture for reducing safety certification cost, but without a substantial change in the

Fig. 1: Stages of the cost model

underlying life-cycle activities. An example where life-cycle activities are introduced can be found in [15] where cost is estimated in several use cases where advanced safety assurance practices are introduced. Their estimations are based on direct values from expert judgement (i.e., interviews). Such approach can be complementary to our cost model to confirm or contrast the results. Their case studies in different industrial domains were exclusively about safety while in our case we focus on safety and security co-engineering practices.

IV. STUDY DESIGN AND METHODOLOGY

Inspired by Runeson and Höst guidelines [16], we present the case study. The goal of our study is to evaluate whether we can measure the integration of safety-security co-engineering practices in a given life-cycle in terms of quality and cost. The hypothesis is that our cost model is sound for this task allowing to take more informed decisions on the final adoption of these practices.

The case study subjects were part of an international consortium investigating co-engineering practices in safety-security critical systems; the EU AQUAS project (Aggregated Quality Assurance for Systems) [3]. The baseline life-cycle of the industrial companies was enriched with new co-engineering practices that were exercised and analysed as pilot projects in their real settings for around three years. The selection of the practices were based on the companies needs or on where the research and tool providers project partners identified that they could provide more potential gain. The EMC was selected given Tecnia's expertise in adapting and using it in other companies [4] and after checking its alignment with the state of the art for critical systems [12], [13]. Thus, for this specific work, the role of Tecnia was to coordinate and validate the data collection from the two industrial case studies and use the EMC. In [4], the five EMC stages presented in Section II were completely considered as part of a complete project, while in this work we replaced the first and second stages with a more lightweight approach to fit in the time and effort planned for the cost analysis. Concretely, we used a Delphi-type approach relying on a panel of experts from the companies. Experts from each of the life-cycle activities were involved, including those that participated in the co-engineering practices during the pilot projects. An expert review was then conducted with four persons from Tecnia to check the global coherence and resolving doubts with the experts about the validity of the collected data.

A. Space case

Thales Alenia Space has more than four decades of experience providing technological solutions for telecommunications, navigation, earth observation, environmental management, exploration, science and orbital infrastructures. The life-cycle under analysis is for an experimental space system including in-flight reconfiguration techniques. The space market is evolving but not at the desired speed in some cases. Notably, the promised benefits of including multi-core processors or System-on-Chip are slow to adopt because of

the strict requirements of safety, security and performance, and the associated validation and certification procedures to comply with the stringent space standards.

Co-engineering practices under study: Two practices are considered. One is to be integrated in the requirements stage and the other in the architecture design stage.

1) Requirements joint review and formal methods (ReqJointReview&FM): A co-engineering meeting among experts from safety and security takes place to discuss about possible interferences between the requirements and creating a matrix of dependencies with criticality classification. In addition, in the context of this project, several timing properties and some static requirements could be verified in the concept phase through formal methods.

2) Safety and Security in the design (SafSecDesign): A combined safety and security component local analysis is performed using component fault trees [17] to model and reason on the interferences caused because of safety and security undesired events. An example on the usage of component local analysis combining safety and security can be found in [18].

B. Medical devices case

RGB Medical Devices, established in 1988, has experience providing medical devices. The kind of project under consideration consists in devices for blood pressure control which operate automatically monitoring the patient and delivering vasoactive drugs to reduce patient's hypertension, and controlling blood pressure. The scenarios include the operating room or intensive care units through pumps trees. This kind of medical device implements a closed loop control system where, after parameterization from the anaesthesiologist, it monitors the patient and keeps his or her state through the infusion pump. Security is increasingly gaining relevance as medical devices are more connected, for instance to the Internet, to networks from hospitals or healthcare organizations, or to other devices or systems. The interest was not on privacy issues, but problems that security vulnerabilities might pose in safety (security-informed safety [2]) and usability issues related to security and safety [6].

Co-engineering practices under study: Two practices are considered. One corresponds to early phases of the life-cycle while the other corresponds to the latest phases.

1) HAZOP analysis for identifying safety/security interactions at the concept stage: Hazard and Operability Analysis (HAZOP) [19] is a technique traditionally used for safety hazards identification. It proposes a systematic approach which is considered effective for multi-disciplinary sessions to identify issues and risks that might not be identified otherwise. HAZOP was used including not only safety experts, but also experts with different knowledge in a unique session, helping to identify interactions of safety with other aspects. A pilot of this practice was conducted with RGB by complementing safety experts with experts on security, medical devices regulations and human-computer interaction [6].

2) Patient model for safety and security testing with hardware-in-the-loop (PModel): Prior to clinical testing, a

patient model, simulating the behavior and conditions of a real patient, is useful to test software, hardware, and communications response of the medical device. A prototype of a patient model was developed and configured to launch tests on different conditions of the patient [6]. The testing includes the validation and verification of safety and security measures.

C. The needed data

First, the life-cycle need to be modelled containing the activities and their flow. Activities that are intended to prevent issues (filters) need to be identified including the new co-engineering practices under study. We used UML activity diagrams which are later extended or annotated with values regarding the introduction of issues and costs. Concretely, for both case studies, the required inputs for the cost model stages 3 (injection table) and 4 (cost model) were:

- UML activity diagram of the life-cycle
- The number of issues injected in each activity
- The cost of applying each filter
- The effectiveness of each filter to detect the issues of each activity
- The cost, for each filter, to fix one issue from each activity
- If applicable due to parallel activity flows, how the issues of the activities are split in the different parallel flows

Several iterations and interviews are conducted for data verification and completeness checks. All data related to issues and cost was requested with optimistic, average and pessimistic values. These three values are useful to create a distribution for the simulation model (this will be explained in Section VI). Experts in each activity of each company were requested to either provide data from the historic of issues or to provide estimations based on expert judgement.

V. OVERVIEW OF THE COLLECTED DATA

We present excerpts of the collected data for the medical devices case. Due to confidentiality we are not allowed to present the whole data nor to present data from the space case except from the results and that their life-cycle was modelled with 11 stages and 30 filters. We use the data codes defined in the enumeration (a to f) in previous Section IV-C.

Regarding (a), Figure 2 shows the UML-like activity diagram representing the life-cycle of the medical device project under study. The activities described in Section IV-B (the co-engineering practices under study) are represented with dashed lines. These two activities, and all the activities with gray color are filters that avoid the propagation of issues. The activities with white background are normal activities where issues can be introduced. As we can observe, all activities are followed by their corresponding revisions and verification. For instance, software development must be carried out in accordance with the requirements of EN 60601-1 and EN 62304. According to these standards, PEMS (Programmable Electromedical System) is defined as a medical device that contains one or more PESS (Programmable Electronic Subsystem), and PESS is defined as a system based on one or more central processing units. Software development is divided into

two stages: Individual software development for each PESS of the device (with its revision and verification filters), and Integration of all PESS that constitute the PEMS software, that is, the device software (and its verification filter).

Regarding the data on issues injected in each activity (b), Table I shows an excerpt. We can observe that the magnitude of the numbers are different for different activities. For example, design issues when architecting the solution can be around 2 issues, while issues in PEMS (e.g., software bugs) can be more numerous (around 30). The issues belong to different activities, thus they have a different nature (e.g., involved assets).

TABLE I: Issues created in the activities (Excerpt 3 out of 21)

	Opti.	Avg.	Pess.
Definition of initial data	12	20	35
High-level Architecture design	1	2	3
PESS Software	9	12	16
PEMS Software	18	30	52

Table II presents an excerpt of the cost to carry out the different filters (c). For the cost model we are interested in the cost of the filters and not in the cost of the activities. As mentioned before, these numbers are based on the historic of the company and expert judgement of the persons involved in these activities within the organization.

TABLE II: Costs of the filters (Excerpt 5 out of 32) in hours

	Opti.	Avg.	Pess.
Revision of initial data	64	96	128
HAZOP Analysis	22	36	60
Revision of architecture	10	24	42
Revision of PESS software	12	16	32
Verification of PESS software	30	37	57
Verification of PEMS software	70	82	114
Patient model verification	120	150	250

Table III presents the effectiveness of the filters in a matrix that relates the filters with the activities (d). This way, we can observe how the “Revision of initial data” is effective in identifying issues from the “Definition of initial data” as it is its main purpose (an average of the 67% of the introduced issues are detected). However, during the “Revision of architecture” it is still possible to identify issues on the “Definition of initial data” that might have escaped from the previous filters and that were noticed later while inspecting other assets. However, in this case, the effectiveness is very limited with only an average of the 6%. Some values are not present because they are not possible according to the activity flow. For instance, the “Revision of initial data” filter cannot do anything with issues from the “High-Level Architecture design” because the architecture design does not exist yet. Also, in some cases it is considered that it is impossible that a filter identify issues from a given activity so the effectiveness is 0% (e.g., “Patient model verification” in identifying issues from the Concept and High-Level design stages).

Coming back to the economic aspects, Table IV presents the costs of fixing issues (e). It relates activities and filters

Fig. 2: Product life-cycle model of the medical device case

TABLE III: Effectiveness of the filters for the types of issues of the activities (Excerpt of a 21x32 diagonal matrix) in %

	Def. of initial data			HL Arch. design			PESS Software			PEMS Software		
	Op.	Av.	Pe.	Op.	Av.	Pe.	Op.	Av.	Pe.	Op.	Av.	Pe.
Revision of initial data	78	67	55									
HAZOP Analysis	45	37	27									
Revision of architecture	8	6	3	72	61	45						
Revision of PESS software	0	0	0	0	0	0	39	28	14			
Verification of PESS software	0	0	0	0	0	0	76	63	45			
Verification of PEMS software	23	16	8	58	45	28	85	72	55	76	64	50
Patient model verification	0	0	0	0	0	0	0	0	0	85	72	51

TABLE IV: Cost to fix an issue detected in a filter and belonging to activities (Excerpt of a 21x32 diagonal matrix) in hours

	Def. of initial data			HL Arch. design			PESS Software			PEMS Software		
	Op.	Av.	Pe.	Op.	Av.	Pe.	Op.	Av.	Pe.	Op.	Av.	Pe.
Revision of initial data	7	10	13									
HAZOP Analysis	10	14	19									
Revision of architecture	8	13	18	1	3	5						
Revision of PESS software							5	8	15			
Verification of PESS software							21	32	53			
Verification of PEMS software	60	84	127	53	74	114	53	74	114	53	74	114
Patient model verification										138	186	273

expressing the cost to fix an issue detected in a given filter and belonging to a given activity. For example, it is almost trivial to fix an issue in the “High-Level Architecture Design” if we identify it during the “Revision of architecture” (around 3 hours). But if we identify the issue later, for example during the “Verification of PEMS software”, the needed changes and their consequences can represent significant rework (an average of 74 hours or more than one hundred in the worst cases).

Regarding parallel activity flows (f), and based on previous experiences, Figure 3 represents the estimation on how issues from the first two stages (Concept and High-Level design) might affect the next activities. For example, we can observe how PEMS software is the activity receiving the highest percentage of issues. Actually, according to the estimations, the two activities from the Software design stage (PESS and PEMS software) together sum to 45%.

Fig. 3: Estimation on the distribution of the issues from the two first stages to the design stage

VI. SIMULATION AND RESULTS

The simulation is an automatic program in which triangular distributions with optimistic, average and pessimistic values have been used to randomly simulate the behavior of the issues that are created in the activities (see Table I), the costs of the filters (see Table II), and the effectiveness of the filters for each type of issue (see Table III). Triangular distributions are continuous probability distributions that are traditionally used when there are minimum, peak, and maximum values, and the actual behavior, or sufficient historical data, are not available to model them. We decided to use triangular distributions but other distributions could have been used. To increase confidence in the simulation outputs, we tried different distributions for the inputs and we confirmed that similar results (i.e., first, second, and third quartiles of the simulation results) are obtained using Weibull, gamma or normal distributions created with those three values.

Then, in the simulation, each injected issue will follow its defined flow and, when reaching a filter, it will be evaluated if the issue will be filtered or not based on the calculated filter effectiveness. If filtered, another triangular distribution will be used for the cost to fix it (see Table IV) and the issue will be removed. The total cost will be the sum of the costs of the filters and the costs of fixing each of the identified issues during the process. The EIC, as defined in Section II, will be the percentage of issues that did not “escape” from the product life-cycle’s filters.

For each analysis, we run the simulation ten thousand times. As an example, Figure 4 presents the results of all the simulations for the baseline scenario and those of introducing the co-engineering practices in the medical devices case. To keep the confidentiality of the data regarding the total cost of projects (for both case studies), we established the origin of the cost (the zero value) in the median of the baseline. This

Fig. 4: EIC and cost boxplots showing the distribution of the simulation results for the medical devices case

way, we can still show the differences in terms of hours from integrating the co-engineering practices.

Instead of using the boxplots for discussion as in Figure 4, we show and discuss the results in a two dimensional space of EIC and cost as we consider that it is more comprehensible and visual to provide information about the cost and quality trade-off. The rectangles in next Figures 5 and 6 correspond to the range of the 25th percentile (P25) and 75th percentile (P75) of both criteria. The smaller rectangles (around the middle of each big rectangle) are the 99% confidence intervals (a statistics metric) of the results of the simulations. Given the large amount of simulations, the averages are very stable so the confidence intervals are small.

Space case: The simulation obtained the results shown in Figure 5. In this case, the quality of the final product is improved if we integrate any of the combined analyses, however, the cost in hours will be higher than the baseline. We can also observe that ReqJointReview&FM, which is to be performed in early stages of the life-cycle, obtains better results than SafSecDesign both in terms of quality of the product and cost. In this situation, it is a business decision if the suggested improvement in quality is enough to introduce a new practice.

The levels of EIC (which are lower than for the medical devices case) can be explained because of the experimental nature of the considered types of projects which can be dif-

ferent from other types of projects within the same company. These preliminary results on the cost model usage are relevant as they show a case which can be considered a normal or intuitive one (i.e., quality is improved but cost is higher).

Medical devices case: Figure 6 shows the simulation results. We can observe that the EIC is already very high for the baseline (a median of 98.91%) which is expected in critical domains with mature life-cycles. That values are even higher when introducing the co-engineering practices. Similar promising results are obtained for the cost where the total number of hours of a project is reduced. It is then suggested that even introducing practices that will consume hours, it pays off both in terms of the total duration of the project and in the chances to have final products with even better quality. Integrating both co-engineering practices dominate the two established criteria. If only a single filter could be applied or we decided to incrementally adopt new practices, the results suggest that we should opt for the application of HAZOP first, since it achieves a better result in EIC and cost than PModel.

Fig. 5: EIC percentage vs Cost in hours for the space case

Fig. 6: EIC percentage vs Cost in hours for the medical devices case

VII. DISCUSSION AND THREATS TO VALIDITY

Our cost model, the EMC, defines a methodological framework. However, the needed data in our analysis was based in some cases on estimations and expert judgement with an acknowledged degree of subjectivity. Our cost model defines stages such as the error recording (see Section II), however the case study stakeholders claimed that currently it is not easy to extract data from real development for each activity and filters, but they still see value in the approach and they made estimations based on evidences and historical data when possible. Stages 1 and 2 of our cost model were covered with a Delphi-like approach to establish an initial behavior of the life-cycle. However, these approaches should be validated with the historical data once they are collected and available in a more systematic way. As summary, the data that we used could have greater dispersion in their estimates than the cases in which the 5 stages are carried out in a systematic fashion. This is an important threat to construct validity. As for the simulation, as mentioned in Section VI, we tried different distributions of the input data (triangular, Weibull, gamma and normal) to confirm that this aspect does not impact our conclusions.

Regarding the generalization of the results and the external validity, a threat is that it is not possible to extrapolate to other organizations/projects. The gathered data are for their type of product and organization, and the results of the co-engineering practices are specific for their case (we explained the study design in Section IV). What it can be generalized is that we have shown that our cost model can be implemented in software-intensive systems of systems companies to investigate the appropriateness of integrating new practices in their current baseline life-cycles.

In the presented approach we are measuring the cost within the life-cycle to deliver the products excluding the system in operation. The cost that might be associated with issues that escaped to all the filters (e.g., the actual exploitation of a vulnerability) and its associated business damage can provide even more business justifications and increase the cost to prohibitive values (e.g., company reputation, penalties).

VIII. CONCLUSIONS

We have focused on safety and security co-engineering practices to provide an economic perspective on the subject. We propose a cost model with a simulation phase to get results in terms of quality and cost. Instead of proposing a new cost model we used the EMC which was already used in non-critical software-intensive systems. Cost models need to be instantiated for the specifics of a company and thus, it is difficult to generalize their results. However, we have shown how it seems beneficial in terms of quality to integrate a set of co-engineering practices in a product life-cycle of two companies, and in one of them it is even profitable in terms of reduction of the global cost of the project. Positive qualitative feedback about the EMC was received from the industrial experts involved in both case studies. They consider the EMC a sound approach to take informed decisions on the adoption of new practices in their life-cycles.

As further work, we will collaborate with other companies from safety-security critical domains to continue exploring and providing positive or negative evidences about the appropriateness of integrating co-engineering practices, and to understand which design choices are in conflict for a specific domain or for a software engineering problem in general.

ACKNOWLEDGMENTS

The research leading to this paper has received funding from the AQUAS project (H2020-ECSEL grant agreement 737475). We would like to thank all the AQUAS partners for their feedback on this work.

REFERENCES

- [1] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 1, pp. 11–33, Jan. 2004.
- [2] K. Netkachova and R. E. Bloomfield, "Security-informed safety," *IEEE Computer*, vol. 49, no. 6, pp. 98–102, 2016.
- [3] L. Pomante, V. Muttillio, B. Křena, T. Vojnar, F. Veljković, P. Magnin, M. Matschnig, B. Fischer, J. Martinez, and T. Gruber, "The AQUAS ECSEL Project Aggregated Quality Assurance for Systems: Co-Engineering Inside and Across the Product Life Cycle," *Microprocessors and Microsystems*, 2019.
- [4] G. Satriani and I. Urretavizcaya, "Do we rework? A path to manage one of the primary cause of uncertainty in software industry," in *QUATIC 2019*, 2019, pp. 179–192.
- [5] A. Ruiz, J. Puellas, J. Martinez, T. Gruber, M. Matschnig, and B. Fischer, "Preliminary safety-security co-engineering process in the industrial automation sector," in *ERTS 2020, Embedded Real Time Systems*, 2020.
- [6] AQUAS project, "D3.2: Combined Safety, Security and Performance Analysis and Assessment Techniques," 2019. [Online]. Available: http://aquas-project.eu/wp-content/uploads/2019/11/D3.2_CombinedAnalysisPreliminary_v2_0bis.pdf
- [7] S. Paul, J. Brunel, L. Rioux, F. Vallée, J. Oliveira, G. Gailliard, J.-L. Gilbert, T. Wiander, M. El Bakkali, A. Fauconney, and D. Chemouil, "Recommendations for Security and Safety Co-engineering - Part A. MERGE project," 2016.
- [8] M. Sango, J. Godot, A. Gonzalez, and R. R. Nolasco, "Model-Based System, Safety and Security Co-Engineering Method and Toolchain for Medical Devices Design," in *DMDC 2019*, 2019.
- [9] B. W. Boehm, "Software Engineering Economics," 1981.
- [10] D. Galin, "Software Quality Assurance: From Theory to Implementation," 2004.
- [11] IEC, "IEC 191-05-24, International Electrotechnical Commission Glossary," 2005.
- [12] NIST, "NIST Planning Report 02-3: The Economic Impacts of Inadequate Infrastructure for Software Testing," 2002.
- [13] SEI, "Trends and New Directions in Software Architecture," 2015. [Online]. Available: https://resources.sei.cmu.edu/asset_files/Webinar/2015_018_100_438676.pdf
- [14] SafeAdapt consortium, "D5.3 Evaluation results of the specified use cases and scenarios," 2016. [Online]. Available: https://www.safeadapt.eu/content/dam/esk/safeadapt/en/documents/SafeAdapt_D5-3_Evaluation_Results_of_the_Specified_Use_Cases_and_Scenarios.pdf
- [15] A. Ruiz, G. Juez, H. Espinoza, J. L. de la Vara, and X. Larrucea, "Reuse of safety certification artefacts across standards and domains: A systematic approach," *Reliability Engineering System Safety*, vol. 158, pp. 153 – 171, 2017, special Sections : Reliability and Safety Certification of Software-Intensive Systems.
- [16] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *EMSE*, vol. 14, no. 2, 2009.
- [17] B. Kaiser, D. Schneider, R. Adler, D. Domis, F. S. Möhrle, A. Berres, M. Zeller, K. Höfig, and M. Rothfelder, "Advances in component fault trees," in *ESREL*, 2018.
- [18] J. Martinez, J. Godot, A. Ruiz, A. Balbis, and R. R. Nolasco, "Safety and security interference analysis in the design stage," in *SAFECOMP 2020 Workshops, DECSoS*, 2020.
- [19] IEC, "IEC 61882: Hazard and operability studies (HAZOP studies) - Application guide," 2016.