Preliminary Safety and Security Co-engineering Process in the Industrial Automation Sector

Alejandra Ruiz[1], Javier Puelles[1], Jabier Martinez[1],
Thomas Gruber[2], Martin Matschnig[3], and Bernhard Fischer[3]

[1] Tecnalia, Derio, Spain
{alejandra.ruiz,javier.puelles,jabier.martinez}@tecnalia.com
[2] AIT Austrian Institute Of Technology, Austria
thomas.gruber@ait.ac.at
[3] Siemens AG, Austria
martin.matschnig@siemens.com; bernhard.bf.fischer@siemens.com

Abstract. The Industrial Automation Sector has a long tradition of showing compliance on functional safety. Ultimately, security was taken into account only at production phase and with a reactive approach. However, this domain is experiencing an increasing need to incorporate cyber-security mechanisms and to provide evidences on security-related standards and applying security by design principles. Both domains have their own regulations defining specific life-cycles. In this work we analyzed IEC 61508 (safety-related) and ISA 62443 (security-related) standards to 1) identify commonalities and create a mapping model, and 2) propose a combined process in the context of safety and security co-engineering. Our approach is qualitatively evaluated by experts on the standards and by practitioners of this domain.

Keywords: Safety · Security · Co-Engineering · Dependability · Quality assurance· IEC 61508 · ISA 62443

1 Introduction

It is becoming increasingly apparent that organizations should look at the safety and security challenges of their systems in a unified way, which is contradictory with a long tradition of safety silos and security silos [19]. Security is usually a requirement in order to ensure safety [18], but different teams are usually responsible for these two tasks separately (one for safety and another one for security). This is justified because of the intrinsic complexity of each area making it difficult to be experts on both. They follow different lifecycles, they need to carry out certification for their own specialized and complex standards, and they use different terminologies complicating their communication.

Combined analysis requires additional effort during early development and it is not certain if it will pay off later. Unfortunately, there are no strong evidences about the cost-effectiveness of dealing with safety and security together. However, safety and security separation led to redundant work and late identification of trade-offs in safety and security requirements [19]. For example, introducing a ciphering security mechanism can increase the time needed to perform a safety-critical system task leading to exceed the worst-case execution time. The motivation of this work is thus to avoid work redundancies optimizing the use of human resources and the time needed while favouring the early identification of potential trade-offs.

This paper is focused on the Industrial Automation sector. Industrial Control Systems (ICS) such as SCADA, Real Time Units, Programmable Logic Controllers or Distributed Control Systems, are connected to hardware and physical systems and they are used to control industrial processes (e.g., processing plants, factories, power plants) and to support key infrastructures (e.g., transportation, airports, seaports and buildings). Failures that end up affecting the ICS may imply not only harming the system itself, or negatively affecting the production or service availability, but also may cause environmental problems or endanger the health and safety of human lives. Examples exploiting security vulnerabilities in safety-critical industrial systems are numerous (e.g., [1,2,3]).

Security concerns are being introduced in the ICS domain, which has been traditionally safety-oriented, focusing on *security informed safety* [18] where security issues could impact the system's overall safety. ICSs are increasingly more software-intensive and "smart" (e.g., ICS are key in the Industry 4.0 paradigm [14]), with needs for data management and communication with external entities. Thus, cyber-security requirements and risks analysis are critical. New challenges arose, next to only considering

security informed safety, privacy issues (a relevant aspect of cyber-security) where safety is not impacted must also be considered (e.g., personal data can be exposed in case of a data breach).

ICSs typically include proprietary software or hardware, as well as communication protocols with limited security. This is mainly because they were designed to meet functional properties in an effective and safe way, and usually overlooking security criteria as they were not traditionally connected with external networks or integrated in a highly-connected and dynamic systems-of-systems context. Besides, the vulnerability of these systems is aggravated due usually to large number of functional entities present in various production installations (e.g., distributed control systems). Also, the locations can be remote and, in some cases, an entity can be even running without human supervision. These entities were usually provided by different vendors, used to have a long operational life, and the intellectual property of the organizations used to be embedded in the deployed systems. Also, most of the industrial processes controlled by ICSs use to be available and functional 24 hours per day, 365 days per year. As summary, the exposure and surface for cyber-attacks on security vulnerabilities on safety-critical ICSs are increasingly high.

The focus of this paper is on a collaborative approach for co-engineering functional safety and cyber-security. We focused on the IEC 61508 international standard entitled "Functional Safety of Electrical/ Electronic/ Programmable Electronic Safety-related Systems" [12], and the ISA 62443 international standards for implementing secure ICSs [13] (more concretely ISA 62443 Part 4.1 "Secure product development life-cycle requirements") were analyzed to identify commonalities and differences between the former (functional safety) and the latter (cyber-security). This in-depth analysis enabled us to propose a combined life-cycle satisfying the needs of the two normative spaces and reducing the risk of duplicated efforts during the certification processes. This first attempt to create a co-engineering process is intended to identify the common links during the project life-cycle and at which phase is most interesting to define interaction points for trade-off discussions. Thus, this process can be used to replace current practices towards a more effective use of the resources when compliance is required for these two widely required standards for ICSs.

This paper is structured as follows: Section 2 presents related works. Section 3 presents background information on the targeted standards. Section 4 presents the methodology and results of the analysis of the standards. Section 5 presents the suggested co-engineering life-cycle. Section 6 presents a qualitative discussion of the results, its strengths and limitations. Finally, Section 7 concludes and presents the further plans.

2 Related work

Some attempts have been made in the past to compare and harmonize different standards analyzing their similarities and divergences. For example, in [4,16,17,21], different functional safety standards were analyzed across different domains where certification is requested. For instance, in [21] a systematic approach is proposed for reusing evidences of a safety-critical component created to comply with a railway's functional safety standard during the compliance process with another standard from the avionics domain. While the focus of trying to reduce redundant work in co-certification is similar; these works focused only on functional safety standards. On the contrary, our focus is on safety and security, requiring a co-engineering process.

According to an international survey [8], managing risks for cyber-security is the activity with more clear priority for companies with ICSs (79% classified it as a main priority and 21 as minor priority). Followed by better complying with regulations, and improving the efficiency on these processes. Regarding this last point, lack of communication between engineering disciplines and their different focus and approaches on security are considered a major issue [8]. Some approaches have studied the idea of combining hazards and threats analysis such as SAHARA (Security-aware Hazard Analysis and Risk Assessment) [15] or FMVEA (Failure Mode Vulnerability and Effect Analysis) [22]. In [23], the idea has progressed and they proposed a combination of different techniques such as Fault Tree Analysis (FTA) [11,20], Stochastic Colored Petri Net (SCPN) [7], Attack Trees Analysis (ATA) and FMVEA for safety-security analysis. Our approach is not to provide a new method for safety and security co-analyses but to propose a combined life-cycle process to reduce the efforts to comply with the standards in isolation and detect the most suitable times for trade-off discussion between the different teams. A related work is also the preliminary co-engineering concepts envisioned for the industrial drives domain [9].

3 Background on the standards under study

3.1 IEC 61508

The IEC 61508 "Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems (E/E/PE, or E/E/PES)" standard [12] is applicable to all kind of programmable electronic safety systems in any industry. It is a general standard proposing a generic approach and it is usually derived for each application domain for specialization. Even though the standard is seen as a regulation for product developers, it is a general regulation that applies to end-users, system integrators and product developers. It is the main standard on the topic of functional safety for ICSs and it defines the requirements of programmable safety systems for the safety of facilities.

The standard covers the whole safety life-cycle, from the initial concept until the system decommissioning or disposal. IEC 61508 proposes three complementary life-cycles. The system level life-cycle can be considered as the leading one. One of its phases, Realisation, is decomposed in two life-cycles which are executed in parallel: The E/E/PE system safety life-cycle (related to hardware) and the software safety life-cycle. IEC 61508 follows a risk-based approach to elicit the safety requirements, however, it does not cover mechanisms that might be necessary to prevent unauthorised persons damaging or affecting the functional safety achieved by the system [10].

3.2 ISA 62243

ISA 62443 [13] has been developed by the ISA99 and IEC committees to improve the security, availability, integrity and confidentiality of the components or systems used in industrial automation and control. If IEC 61508 standard is considered the reference standard for functional safety, similarly it is expected to happen with ISA 62443 for cyber security. At the moment of writing this paper, not all the parts that will be part of the ISA 62443 series are published and some are still considered draft.

The ISA 62443 clearly identifies three main roles and their compliance responsibilities: the asset owner, the system integrator and the product supplier. The standard only addresses the development of the components and system which are part of an IACS (Industrial Automation and Control System) but does not cover the design, installation or operation of that IACS. The standard distinguishes between the product development process (ISA 62443-4-1) and the system or subsystems integration process performed by the system integrator (ISA 62443-2-4) with an acquisition perspective where components might be created by different suppliers and an integrator needs to ensure security when components work together after the integration. The standard also includes security measures as well as security conformance metrics in other parts.

4 Standards analysis

We first present the methodology for the standards analysis in Section 4.1 and then in Section 4.2 we discuss and present examples of the identified commonalities and differences.

4.1 Methodology

For the two standards analyses, two standardization experts lead the work. One of the experts has a long experience as consultant for standards appliance, specially on process improvement and on domains related to critical infrastructures. The other expert has a strong background on functional safety standards evolution and related research. Both of them have cyber-security knowledge. The work was performed during six weeks. The objective was to identify the commonalities (activities, requirements, roles, evidences, techniques, methods) between both standards.

The work was done in an iterative way. First a study of the standards was performed followed by initial joint sessions for discussing the standards regarding their structure, scope, techniques, management approaches etc. Then, the experts conducted two workshops with industrial partners with a special focus on security aspects. One of the discussions was on how the ISA 62443-4-3 was applied in a specific project. This part of the standard deals with precise technical requirements. The second discussion was on how the process of a company was being defined to apply ISA 62443 in a systematic way. After these two workshops, the interaction continued with focused discussions on the objectives of each practice (both from ISA 62443 Part 4.1 and IEC 61508). As a result of these discussions the mappings between the practices were identified.

4.2 Equivalence mappings

The IEC 61508 and ISA 62443 standards have been analyzed in order to search for equivalences between the different standards requirements. Even though we knew that each of the standards has a particular scope as described in Section 3, we assumed a scenario where a functional safety standard is already in place and there is a need to expand the compliance process to comply with cyber-security regulations. This assumption is aligned with the safety-oriented tradition of ICSs where security concerns arrived later. Thus, in our study the identified equivalences are true in one direction and we did not check whether the equivalences are bidirectional. For example, an identified equivalence can be related to the fact that the security requirement is conceptually contained in the safety requirement but, this does not mean that satisfying only the security requirement is enough to satisfy the safety one.

In the beginning, equivalences were sought for the management of cyber-security of a component (ISA 62443-4-1) with the management of the functional safety of an electronic device (IEC 61508). When looking for equivalences, it has been conceptually assumed that when we talk about functional safety we also include cyber-security that affects functional safety.

Table 1 shows two examples of what we called "a full equivalence mapping", that is a requirement identified in both standards which can be considered compatible. Thus, once you provide evidence for compliance for IEC 61508, same evidence can be used to prove compliance with ISA 62443. This is the case of the requirement dealing with identifying people responsible of the tasks. This is requested in

Table 1. Examples of equivalent process requirements

ISA 62443 Cyber security	**IEC 61508 Functional safety**
Part 4-1 Practice 1 Security management. SM2: Identification of responsibilities A process shall be employed that identifies the organizational roles and personnel responsible for duties for each of the processes required by this standard.	***Part 1 - 6.2.1 Requirement*** An organisation with responsibility for an E/E/PE safety-related system, or for one or more phases of the overall, E/E/PE system or software safety life cycle, shall appoint one or more persons to take overall responsibility for: the system and for its life cycle phases; coordinating the safety-related activities carried out in those phases; *(many other items were not included for space limitations)*
Part 4-1 Practice 1 Security management. SM1: Development process A general product development/maintenance/support process shall be documented and enforced that is consistent and integrated with commonly accepted product development processes (for example, ISO 9001 certified processes) that include, but are not limited to: – configuration management with change permission controls and audit record logging; – product description and requirements definition with requirements traceability; – software or hardware design and implementation practices, such as modular design; – repeatable testing verification and validation process; – review and approval of all development process artifacts; and – life-cycle support.	***Part 1 - 6.2.5 Requirement*** Procedures shall be developed for ensuring prompt follow-up and satisfactory resolution of recommendations relating to E/E/PE safety-related systems, including those arising from: – hazard and risk analysis (see 7.4); – functional safety assessment (see Clause 8); – verification activities (see 7.18); – validation activities (see 7.8 and 7.14); – configuration management (see 6.2.10, 7.16, IEC 61508-2 and IEC 61508-3); – incident reporting and analysis (see 6.2.6).
Part 4-1 Practice 1 Security management. SM4: Security Expertise process A process shall be employed for defining security training and assessment programs to ensure that personnel assigned to the organizational roles and duties specified in 6.3, SM2 - Identification of responsibilities, have demonstrated security expertise appropriate for those processes.	***Part 1 - Requirements:*** – 6.2.3, – 6.2.12, – 6.2.13, – 6.2.14, – 6.2.15, – 6.2.16

both standards. Another example is the development process requirement; while the fist example is quite straightforward, in this second requirement example we have taken into account how the development process is applied in both cases. For instance, in the IEC 61508 the configuration management should be applied to the different artefacts requested in the development so the system description, the hardware, the software designs and all the validation and testing results are included. Also, the validation and verification plans should be systematic, repeatable and traceable. The quantity of documentation and evidences required for the development process is a good example of how relevant is to avoid this redundant work. In some cases the equivalence is not a one-to-one mapping of two articles or statements but one-to-many. This is the case of the last example included in Table 1 where the security expertise process is not clearly identified in one of the requirements of IEC 61508 but in six of them.

During the analysis it was revealed that the mentioned "full equivalence map" is not frequent and the common situation was "a partial equivalence map" as the examples presented on Table 2. In this case, the requirements can be found in both standards but they cannot be considered as being completely equal. When looking at these two examples (process verification and third-party suppliers support), we can identify parallel process activities where safety and security could interact and discuss trade-off decisions as each of the requirements is focusing either on safety or on security. It might be considered as a unique process activity and equivalent only if the deployed product development process has considered co-engineering practices.

Table 2. Example of a partially equivalent process requirements

ISA 62443 Cyber security	**IEC 61508 Functional safety**
Part 4-1 Practice 1 Security management. SM11: Process verification A process shall be employed for verifying that, prior to product release, all security-related processes required by this specification have been carried out.	***Part 2 - 7.9.2 and Part 3 - 7.9.2 Requirements*** P2-7.9.2.1 The verification of the E/E/PE safety-related systems shall be planned concurrently with the development (see 7.4), for each phase of the E/E/PE system safety lifecycle, and shall be documented. P3-7.9.2.1 The verification of software shall be planned (see 7.3) concurrently with the development, for each phase of the software safety lifecycle, and shall be documented.
SM8: Third party embedded component security A process shall be employed to identify and manage the security risks of all externally provided components used within the product.	***Part 1 - 6.2.17 Requirement*** Suppliers providing products or services to an organization having overall responsibility for one or more phases of the overall, E/E/PE system or software safety life-cycles, shall deliver products or services as specified by that organization and shall have an appropriate quality management system.
SM7: Development environmental security A process that includes procedural and technical controls shall be employed for protecting the integrity of the development environment, production and delivery, including private keys, and the design, implementation and release of a product or product update (patch).	***Part 1 - 6.2.3 c) Requirement*** Software configuration management shall [...] maintain accurately and with unique identification all configuration items which are necessary to meet the safety integrity requirements of the E/E/PE safety related system. Configuration items include at least the following: safety analysis and requirements; software specification and design documents; software source code modules; test plans and results; verification documents; pre-existing software elements and packages which are to be incorporated into the E/E/PE safety related systems; all tools and development environment which are used to create or test or carry out any action on the software of the E/E/PE safety related system.

5 Proposed co-engineering process

As a result of the mapping study presented in Section 4, we propose the definition of a co-engineering process where full equivalence requirements will be taken into account just once, and the partially equivalent ones will be integrated in a unique activity that consider both perspectives. This preliminary co-engineering process is created with the aim to identify common links during the project life-cycle and at which phase it is most interesting to define interaction points for trade-off discussions. Table 3 presents the links of the different life-cycle phases of our proposed life-cycle regarding the related standards' parts and articles. This mapping can be used by practitioners to reference to the specific requirements in each of the phases.

The proposed life-cycle contains a set of phases. We include a brief summary of them below but the details can be found in the referenced standards' parts and articles of Table 3. We designed our co-engineering process based upon the well-known V-model [5] inspired by the implicit V-model of IEC 61508 with the idea to increase the comprehension of the concepts (see Figure 1). The dark gray V represents the system life-cycle and the light gray V's represent the software life-cycle (left) and the hardware life-cycle (right) which need to be synchronized. The arrow between the V's represent the validation direction where the final hardware validation is performed once the software has been deployed in the specific hardware.

- Development Management: This activity is performed along the development life-cycle as functional safety management and security management should be coordinated.
- Concept: This is mainly a safety phase where the concept of the system is defined.
- Overall scope definition: The basic blocks and approaches for safety and security should be agreed.
- Overall requirements allocation: In this phase the requirements to fulfil the safety concepts and security concepts should be derived into requirements and allocated to blocks of the future system design.
- System requirement specification: System requirements are agreed from safety and security perspectives.
- Software requirement specification: This phase focused on the software and special attention should be given to reduce security vulnerabilities and eliminate possible security breaches.
- Validation planning: In this phase validation should be planned from the functional perspective but also ensuring that hazards and threats are sufficiently mitigated or deleted.
- Software architecture: This phase is focused on the architecture definition and both safety and security mechanisms should be allocated to the different components of the architecture.
- Software system design: This phase is mainly safety-driven software design.
- Coding: This phase should be security-driven. Although best practices for coding and coding standards should be followed, security practices should be prioritised to avoid vulnerabilities.
- Module testing: The different modules of the system should be tested to ensure the fulfilment of the safety and security requirements.

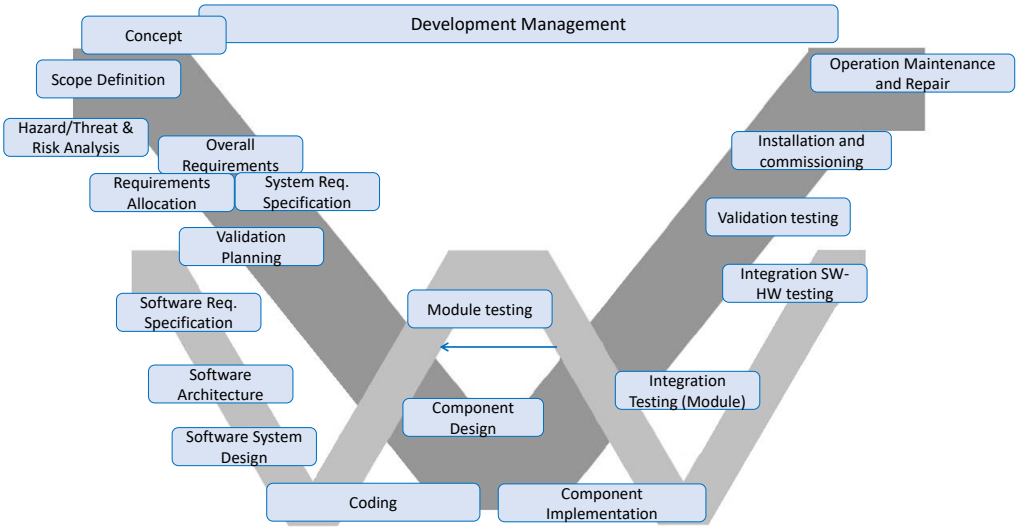

Fig. 1. The V-shaped proposed co-engineering process

Table 3. Co-Engineering life-cycle with corresponding standards references

Life-cycle phase	IEC 61508 Functional safety	ISA 62443 Cyber security
Development Management	*Part 1 6 Management of functional safety*	*Part 4-1 Practice 1: SM1, SM2, SM3, SM4, SM5*
Concept	*Part 1 7.2 Concept*	
Overall scope definition	*Part 1 7.3 Overall scope definition*	*Part 4-1 Practice 2: SR1*
Hazard/Threat and Risk Analysis	*Part 1 7.4 Hazard and risk analysis*	*Part 4-1 Practice 2: SR2; Part 4-1 Practice 3: SD4, SD5; Part 4-1 Practice 5: SV3*
Overall requirements	*Part 1 7.5 Overall safety requirements*	*Part 4-1 Practice 2: SR3, Part 4-1 Practice 3: SD5; Part 4-1 Practice 8: SG1, SG2*
Overall Requirements Allocation	*Part 1 7.6 Overall safety Requirements Allocation*	*Part 4-1 Practice 2: SR3*
System requirements Specification	*Part 1 7.10 E/E/PE system safety requirements specification*	*Part 4-1 Practice 2: SR4*
Software Requirement Specification	*Part 3 7.2 Software safety requirements specification*	*Part 4-1 Practice 2: SR4*
Validation planning	*Part 1 7.8 Overall safety validation planning Part 3 7.3 Validation Plan for SW aspects of system safety*	*Part 4-1 Practice 2: SR5 Part 4-1 Practice 3: SD3 Part 4-1 Practice 5: SV2, SV5*
Software Architecture	*Part 3 7.4.2 SW design and development. General Requirements Part 3 7.4.3 SW design and development. Requirements for SW Architecture design Part 3 7.4.4 SW design and development. Programming languages*	*Part 4-1 Practice 1: SM7 Part 4-1 Practice 3: SD1, SD2, SD6 Part 4-1 Practice 4: SI4*
Software System Design	*Part 3 7.4.5 SW design and development. Requirements for detailed designed - SW system design*	
Coding	*Part 3 7.4.6 requirements for code implementation*	*Part 4-1 Practice 1: SM6, SM7*
Module Testing	*Part 3 7.4.7 Requirements for SW module testing*	*Part 4-1 Practice 4: SI1, SI2, SI3*
Integration Testing (Module)	*Part 3 7.4.8 Requirements for SW integration testing*	*Part 4-1 Practice 4: SI2, SI3*
Integration Testing (components, subsystems and programmable electronics)	*Part 3 7.5 Programmable electronics integration (HW-SW)*	*Part 4-1 Practice 4: C17SI2, SI3*
Validation testing	*Part 3 7.7 SW aspects of of system safety validation*	*Part 4-1 Practice 5: SV3, SV4*
Overall installation and commissioning	*Part 1 7.14 Overall installation and commissioning*	*Part 4-1 Practice 8: SG1, SG2, SG3. SG4,SG5, SG6, SG7*
Operation, maintenance and repair	*Part 1 7.15 overall operation, maintenance and repair*	*Part 4-1 Practice 6: DM1, DM2, DM3, DM4, DM5, DM6 Part 4-1 Practice 7: PM1, PM2, PM3, PM4, PM5*

- Integration testing (module): In this phase it is important to ensure that no emergent behaviour appears which could affect to safety or security. Both properties should be simultaneously taken into consideration.
- Integration testing (components, subsystems and programmable electronics): Similarly to previous phase but considering the deployment of the software into the hardware.
- Validation testing: Validation should ensure that safety and security mechanisms and mitigation strategies are working as expected.
- Overall installation and commissioning: It is important to take into account in this phase that no security breaches are introduced while ensuring safety.
- Operation, maintenance and repair: Each modification done to the system is agreed between safety and security teams so no new vulnerabilities or hazards could be introduced.

6 Qualitative evaluation and discussion

This section presents the conclusions of three industrial and standardization experts after presenting them our results and the proposed life-cycle. This qualitative discussion is an evaluation on the soundness of our approach.

First, they wanted to emphasize the relevance of the objective and motivation presented in Section 1. In the industrial context, where time and money are of utmost importance, combining safety and security in a single process has the potential to save a lot of effort by applying activities that satisfy both, safety and security. Though they are in favor of combined methods, they acknowledge that there are arguments, in particular from practitioners in industry with a traditional mindset, why treating safety and security simultaneously can create difficulties. One of those arguments is that safety experts and security experts mostly "think in a different manner" (i.e., solve their problems according to their own paradigms and, thus, have difficulties to follow the arguments of the experts of the other discipline or are not willing to do this). Another argument is that standardization for the two quality attributes is still widely separate. So these safety and security experts prefer separating the development to address both aspects independently. The problem with the separate approach is, however, that realizing safety functions and security controls independently may cause double work because the required functions are partly the same. Even worse, the solutions resulting from independent safety and security engineering flows are likely to be incompatible due to mutual influences not being treated. The necessary iterations for removing redundancies or correcting the incompatibilities are costly. Therefore the three experts agree that the conducted standards analysis to systematically identify mappings is a correct approach.

Industry shrinks back from investing effort into training, which is doubtlessly necessary when introducing the new co-engineering processes. However, on the long run, the savings from combined processes with less iterations will prevail over the investment in training and the additional costs for purchasing or licensing advanced co-engineering tools. Moreover, rising complexity of future systems should even increase the long-term advantage. It is also important to recognize that co-engineering can be introduced for individual phases and, thus, step-by-step. In the following, we discuss in more detail the effectiveness of the approach.

For the proposed co-engineering, one example for synergistic activities is the fulfillment of integrity requirements. Integrity requirements in the safety context mean that sensor data must not be corrupted before being processed, whereas in the security domain integrity is focused towards data not being willingly manipulated to damage the system's operation. Integrity is a requirements set that is present in both, ISA 62443 and IEC 61508. By only looking at the safety side, a simple CRC (Cyclic Redundancy Code) calculated over payload and attached to data packets is sufficient for most cases. However, security attacks are able to manipulate that data including the CRC. In this case, introducing a MAC (Message Authentication Code) prevents that data from being changed maliciously and ensures that a change of data can be detected, thus enabling the system to take safety measures (for example introducing redundancies in combination with an adjudicator can satisfy safety and security requirements [6]). Implementing security measures always seem to be an expensive investment dealing with unknowns and with little visibility of its benefits (only when security measures fail). With a combined co-engineering process, some of these efforts can be saved due to the synergies emerged by common process activities that may result in a partial design solutions that can fulfill many requirements at once (instead of creating one design solution for a single requirement). One example for such a process activity is interference analysis where requirements are analyzed for their interference on each other. This requires that quality attribute

requirements (such as safety and security) have to be analyzed together in a common process. The goal of interference analysis is to find requirements that are influenced by, or have commonalities with others. In this case, the analysis of the safety data integrity requirement and the security data requirement result in a single solution that satisfies both requirements (MAC).

Security can hardly be measured and modelled statistically, which prevents safety methods being adopted as they are into the security domain, i.e. FTA or Failure Mode and Effects Analysis (FMEA) have to be refined and extended for security analysis (FMVEA). Thus, established safety analysis methods can not be re-used in a synergistic way. Also, one major problem of safety-related systems arises if software has to be updated. This leads to additional efforts for maintaining the (already gained) certifications. Since modern Industrial Automation Control Systems (IACS) have interfaces for external data exchange and software/systems are never truly secure, updates must be implemented on a regular basis. The proposed combined co-engineering flow eases re-certification efforts, considering the example above, only the functionality of the MAC (which is able to provide data integrity handling and additionally indication of data integrity and authentication) has to be proven again since the MAC satisfies both, the safety integrity requirement and the security integrity requirement.

Overall, modern ICSs all need safety and security measures. There might only be some rare cases (very small systems) where safety can be implemented encapsulated and not being subject to security issues. The proposed life-cycle, which emerges from the analysis of commonalities and differences of the standards, provides guidelines for cost-effective co-engineering in ICSs.

7 Conclusions

We have shown the feasibility to define a development and co-certification life-cycle for functional safety and security. In this preliminary work, we aligned the phases of the life-cycle identifying common link points. We focused on IEC 61508 and ISA 62443 to propose a co-engineering solution for the Industrial Automation sector. The presented discussions from practitioners help to understand the approach benefits (e.g., higher confidence of a correct design gained by combined analysis in early design stages) and limitations (e.g., initial effort of setting up the new co-engineering flow, higher effort and involvement of experts in early life-cycle stages). As further work, we aim to refine the process with empirical data of its usage (at least a part of it), as well as proposing and integrating advanced co-engineering practices (e.g., safety-security interaction points and interference analysis [19]) in the definition of the process.

Acknowledgments: The research leading to this paper has received funding from the AQUAS project (H2020-ECSEL grant agreement 737475).

References

1. arstechnica: Hackers trigger yet another power outage in ukraine, ars TECHNICA (2017), `https://arstechnica.com/information-technology/2017/01/the-new-normal-yet-another-hacker-caused-power-outage-hits-ukraine/`
2. BBC: report 'timeline: How stuxnet attacked a nuclear plant' (2010), `https://www.bbc.com/timelines/zc6fbk7`
3. BBC: report hack attack causes 'massive damage' at steel works (2014), `http://www.bbc.com/news/technology-30575104`
4. Blanquart, J.P., Astruc, J.P., Baufreton, P., Boulanger, J.L., Delseny, H., Gassino, J., Ladier, G., Ledinot, E., Leeman, M., Machrouh, J., Quéré, P., Ricque, B.: Criticality categories across safety standards in different domains. In: Embedded Real Time Software and Systems (ERTS) (2012)
5. Forsberg, K., Mooz, H.: The relationship of system engineering to the project cycle. In: Proceedings of the National Council for Systems Engineering First Annual Conference. pp. 57–61 (1991)
6. Gashi, I., Povyakalo, A., Strigini, L., Matschnig, M., Hinterstoisser, T., Fischer, B.: Diversity for safety and security in embedded systems. In: IEEE Int. Conf. on Dependable Systems and Networks (2014)
7. Gehlot, V., Nigro, C.: An introduction to systems modeling and simulation with colored petri nets. In: Proceedings of the Winter Simulation Conference. pp. 104–118. WSC '10, Winter Simulation Conference (2010)
8. ARC advisory group, Kaspersky, T.M.: The state of industrial cybersecurity 2019 (2019)
9. Gruber, T., Schmittner, C., Matschnig, M., Fischer, B.: Co-engineering-in-the-loop. In: Computer Safety, Reliability, and Security. pp. 151–163 (2018)
10. IEC: Functional safety and iec 61508. a basic guide (2004)

11. IEC 61025: Fault tree analysis, second edition (2006)
12. IEC 61508:2010: Functional safety of electrical/electronic/programmable electronic safety-related systems (2010)
13. IEC 62443: Industrial communication networks network and system security (2010)
14. Lu, Y.: Industry 4.0: A survey on technologies, applications and open research issues. Journal of Industrial Information Integration **6**, 1 – 10 (2017)
15. Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: Sahara: A security-aware hazard and risk analysis method. In: Design, Automation Test in Europe Conference Exhibition (DATE). pp. 621–624 (2015)
16. Machrouh, J., Blanquart, J.P., Baufreton, P., Boulanger, J.L., Delseny, H., Gassino, J., Ladier, G., Ledinot, E., Leeman, M., Astruc, J.M., Quéré, P., Ricque, B., Deleuze, G.: A cross-domain comparison of software development assurance standards. In: Embedded Real Time Software and Systems (ERTS) (2012)
17. Machrouh, J., Blanquart, J.P., Baufreton, P., Boulanger, J.L., Delseny, H., Gassino, J., Ladier, G., Ledinot, E., Leeman, M., Astruc, J.M., Quéré, P., Ricque, B., Deleuze, G.: Cross domain comparison of System Assurance. In: Embedded Real Time Software and Systems (ERTS) (2012)
18. Netkachova, K., Bloomfield, R.E.: Security-informed safety. IEEE Computer **49**(6), 98–102 (2016)
19. Pomante, L., Muttillo, V., Kena, B., Vojnar, T., Veljkovi, F., Magnin, P., Matschnig, M., Fischer, B., Martinez, J., Gruber, T.: The AQUAS ECSEL Project Aggregated Quality Assurance for Systems: Co-Engineering Inside and Across the Product Life Cycle. Microprocessors and Microsystems (2019)
20. Ruijters, E., Stoelinga, M.: Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. Computer Science Review **15**, 29–62 (2015)
21. Ruiz, A., Juez, G., Espinoza, H., de la Vara, J.L., Larrucea, X.: Reuse of safety certification artefacts across standards and domains: A systematic approach. Reliability Engineering & System Safety **158**, 153 – 171 (2017)
22. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (fmea). In: Computer Safety, Reliability, and Security. pp. 310–325 (2014)
23. Verma, S., Gruber, T., Schmittner, C., Puschner, P.: Combined approach for safety and security. In: Computer Safety, Reliability, and Security. pp. 87–101 (2019)