A SURVEY ON TECHNIQUES REQUIREMENTS FOR INTEGRATING SAFETY AND SECURITY ENGINEERING FOR CYBER-PHYSICAL SYSTEMS

MOHAMMED F. H. ABULAMDDI

Department of Software Engineering, University of Palestine, Palestine

ABSTRACT

Nowadays, safety and security have become a requirement, integrated to each other, for information systems as a new generation of infrastructure systems distributed throughout networks. That opened the door for questions on whether these systems are safety-critical especially since they were tested in a closed, separated environment and are now deployed in an uncontrollable environment, namely the internet, where the number of threats is enormous. So it opened the door to talk about new development approach methods that take safety and security into consideration during the system development life cycle and most importantly, identifying hazard, risks and threats.

We will conduct a survey exploring technical languages that were created by the scholars to combine safety and security requirement engineering and accident analysis technique languages.

KEYWORDS

Dependability Requirements, Safety requirements, Security requirement, Cyber-Physical Systems, Accident Analysis Technique.

1. INTRODUCTION

The fields of safety and security use different development tools and methods. Consequently, these two separate but related research areas utilise different concepts, tools and methods.

The following analogy is commonly used to illustrate the difference between the fields of safety and security: a safety engineer believes that it is important for buildings to have easy accessible exit in case of emergency, whereas a security engineer sees the emergency door as a loophole that can provide access to the building to unauthorized personnel and therefore it must be secured. But, there is also another important distinction between the two: 'Security is concerned with the risks originating from the environment and potentially impacting the system, whereas safety deals with hazards arising from the system and potentially impacting the environment' [1]. It is important, during system development and operations, to identify, analyse, evaluate and finally deal with as many relevant risks as possible. At the same time different techniques are used within the fields, specifically because safety deals with unintentional hazards and security with intentional threats.

The safety techniques used to discover threats to a system are not the problem; problems arise when a system is connected to the internet and becomes subject to the rules of another environment that is open, which completely opposite to the closed environment for which the system was designed. Security and safety engineers usually work independently and employ different methods when developing a system: security engineers do not take in consideration safety aspects and safety engineers do not consider security. But, it makes little sense to invest effort in ensuring the dependability of a system while ignoring the possibility of security vulnerabilities. A basic level of security - in the sense that a software system behaves properly even in the presence of hostile inputs from its environment - should be required for any software system that is connected to the network, and used to process sensitive or personal data, or used by an organization for its critical business or operational functions.

Safety and security models and tools have been under focus from different perspectives. Some researchers focused on the architectural framework while others focused on narrowing down the gap between the definitions and terminology adaptation in both safety and security or narrowing down techniques requirements and tools used in the system development life cycle. It is important to note that each and every technique built and used in a specific industry has its own threat analysis and mathematical formulas, even if they all under the safety engineering umbrella. Furthermore each of these techniques was built and used according to technical reports unlike how it is done in security engineering, which can lead to issues especially since these techniques are separated from their environments. So, potential risks might arise when these systems are functioning within their environments.

2. RELATED WORK

In the studies conducted by Benjamin [2], researchers focused on security in a try to build a conceptual framework that deals with definitions and terminology related to security requirements from the beginning of the system development life cycle especially during the elicitation phase and analysis requirement which is reflected on the conceptual framework that the researches put as a base for comparison and focused on analysing the methods used like: Common Criteria, Secure Tropos, ISSRM, SREP, MSRA, Problem Frames, and the methods that depend on UML to the extent that these methods provide coverage in relation of the system development life cycle when used.

Eames and Eames [3] surveyed the nature of integration whether it is consolidated, combined, synthesised, unified, or harmonised. The research addresses each of these states separately and the definition of each one of them when implemented on safety and security. Furthermore, the researches focused on the interaction requirements and proposed enhancing traceability in documentation using a case study on Military Air Traffic Control. System was addressed from both the safety and security perspectives and their respective techniques that separately were undertaken by two separate teams each serving a perspective. The integration between the two, however, took place during the case documentation phase through conflict resolution and integration of requirements.

In the studies conducted by Firesmith [4] [5], he focused on developing the definitions for safety and security domains and comparing them to one another and to survivability engineering. He also created a unified definition that includes safety, security, and survivability engineering called defensibility and then created information models using UML class and founded relationships and definitions between safety engineering and security engineering.

Allenby and Kelly [6] described the technique for eliciting and analysing functional safety requirement for aircraft engines by using scenario and deriving Use-Case (UC) and hazard analysis by addressing HAZOP and putting guidewords that comply with the UC, and then integrating HAZOP and scenario-base requirement leading to the conclusion that it is possible to limit mitigating risks.

Srivatanakul et al. [7] worked on creating HAZOP-Based security analysis on misuse case model through extending guideword notation that is used in the HAZOP and merged its use with Misuse-Case UML notation.

Alexander [8] has also addressed the use of Misuse-Case (MUC) and Failure Mode and Effect Analysis (FMEA) by facilitating the use of MUC during the first phase because the analysis that resulted from it, defines the risks and threats that might face the system and the pieces of information resulting from MUC are used as input for FMEA analysis. He also addressed the interplay of UC and MUC with functional and non-functional requirements in details.

In 2005, Zafar and Dromey applied the behavioral tree formalism and genetic software engineering approaches to deal with safety and security properties in system design [9]. In 2009, Sun et al. used the Maude rewriting logic formalism to automatically spot contradictory requirements between safety and security, for a very basic Use-Cases[10].

Sommerville [11] had created a new notion called *Concerns* which was the motive behind creating this notion and helped in having improved processing for eliciting and analysing safety requirements. Furthermore, this is also important in avoiding conflicts in requirements since notion concerns are basically requirements and organizational goals cross-checking which were applied on a case study.

Tor and Guttorm [12] have prepared for an experimental comparison between Use-Case diagrams and textual Use-Cases in defining safety hazard identification. Their experiment concludes that by using the textual use cases, they were able to identify more failure modes or threats than using case diagrams.

Tor et al. [13] have conducted two separate experiments to compare between sequence diagrams and textual use cases in hazard identification to find out which is more appropriate in discovering risks that might appear during the early stages of the system development life cycle. They concluded that sequence diagrams are better for the identification of hazards than textual use cases.

Cambacédès and Bouissou [14] focused on finding new methods to deal with modeling safety and security interdependencies with Boolean logic Driven Markov Processes (BDMP), a technique that depends on graphical modeling and mathematical formalism. However, using this newly founded method is impractical because it requires knowledge and hands-on experience as it is much similar to attach tree and fault-tree. The newly founded technique was derived from a real case study used [15] where the focus was on modeling the Stuxnet Attack with BDMP hoping towards more formal risk assessments.

In the field of graphical modeling, the work of Fovino and Masera [16] can be seen as relatively close to the BDMP. In fact, Fovino's work is based on a combination of classical fault trees and attack trees, which are static structures and do not provide advanced automatic treatment capacities of BDMPs.

In the study conducted by Raspotnig et al. [17], the focus was on the techniques used in risk definition in safety and those used in defining risks in security and worked on defining each category of techniques separately. This study represents in-depth study on each technique. It also points out its weaknesses and strengths and in which phases of a system development life cycle, each technique could be used.
In this article, we will conduct a survey according to the techniques requirments that combine safety and security.

3. SAFETY AND SECURITY TECHNIQUE REQUIREMENTS LANGUAGES

There are many security technique languages that are used to analysis the risks during the system development stage .Also for safety, there are many technique languages that are used for hazard analysis. In this section, we will only handle the technique languages that combine safety and security together.

3.1 HAZOP - The HAZard and Operability

HAZOP, as its name suggests, is a technique for identifying and analyzing hazards and operational threats in a system. The technique originates from the chemical industry in the 1970s, but has been applied in different types of industries since that time. The ability of this technique is to be adapted to many types of systems taking different parts into consideration such as hardware equipment, software, procedures and humans. It is sought to be one of the reasons why this technique is popular. An important part of the technique is the use and combination of parameters and guidewords for the risk identification, such as the parameter flow combined with the guidewords: more, less or other than. There are a few examples of applying HAZOP in the security field [18], but with specialized post- and pre-guidewords combined with attributes for security considerations, e.g., *pre-guide-word*, deliberately combined with the attribute manipulation and post-guide-word insider. HAZOP is used with a worksheet recording:

a. *Item* – a description of the function or purpose of item that is analyzed
b. *Parameter* – interaction description between the components or design intent of a component, affecting its operation, for example flow of data, isolate, absorb, start-up, etc...
c. *Guide-word* – the word combined with the two items above, to stimulate creativity among participants for imagining design deviation, e.g., no, less, other, early. (Table 2) shows guideword interpretations for attributes of Messages.
d. *Consequence* – a description of the deviation from design intent found with guideword
e. *Cause* – possible causal factor descriptions for the deviation found
f. *Hazard* – a description of the hazard and associated risk that occurs from the consequence description above
g. *Recommendations* – a description of the recommended mitigations for the hazard

Diagrams are developed in early phases of the development, typically in an early design phase. The technique is also well suited for usage in the detailed design phase. The technique is used by safety personnel, and it is important that the process is led by an experienced team leader. When applying the technique, safety and system experts, developers, and users normally give input. HAZOP is one of the most popular generic techniques used for hazards identification. Although the technique was originally developed for chemical plants and processing systems, it has also been applied to a range of systems such as extensive chemical processing plants, large distributed defense systems and embedded aircraft braking systems [19], it has been used with different systems within various industries. Some parts of the technique are however modified for special purposes or to work with particular systems especially guidewords and the parameters for security [18]. As mentioned above, there is a range of application areas for HAZOP. It is assumed that the technique can be applied both to safety critical systems, such as reactor protection systems, and to safety related systems. There have also been examples of using the HAZOP within the security field [18], in a security critical context by adding guideword to security risk analysis.

Table 2. Example of Suggested guideword interpretations for Messages attributes[20].

Entity=Message		
Attribute	**Guide word**	**Interpretation**
predecessor/ successor	*No*	Message is not sent when it should be.
	Other than	Message sent at a wrong time.
	As well as	Message sent at a correct time and also at an incorrect time.
	Sooner	Message sent earlier within message sequence than intended.
	Later	Message sent later within message sequence than intended.

sender/ receiver	*No*	Message is not sent when intended (to any destination).
	Other than	Message sent to a wrong object.
	As well as	Message sent to a correct object and also an incorrect object.
	Reverse	Source and destination objects are reversed.
	More	Message sent to more objects than intended.
	Less	Message sent to fewer objects than intended.

The recommended steps in a HAZOP study, which is based on examining design representations of a system, are: identifying each entity in the design representation; descripting the interaction between the components of a component affecting its operation like flow of data; applying guidewords to attributes by investigating deviations from the design; investigating the causes and consequences of each deviation; and descripting the recommended mitigations for the hazard/risk.

3.2 BDMP - Boolean Logic Driven Markov Processes

BDMP technique several researches [21] focused on finding new methods to deal with modeling safety and security interdependencies with BDMP. A technique depends on graphical modeling and mathematical formalism (Figure 1). However, using this newly founded method is impractical because it requires knowledge and hands-on experience because it is very much similar to attack tree and fault-tree with Markov processes.

The ability to formulate BDMP enables modelling dynamic feature with triggers. BDMP is used to model the different combinations of events that may lead to undesired events, such as system. In a tree, these events represent the leaves. Each leaf is associated to a '*triggered Markov process*' that models its different states. This process can be in a '*required*' and '*Not-Required*' mode or in an '*Idle*' or '*Active*' mode for safety-related and security-related leaves respectively. This method, besides other outputs, gives quantitative results including the sequences that most probable lead to unwanted events.

BDMP is suitable for risk evaluation process and it consists of three phases:

a. *Context definition* defines the scope and boundaries of a system and the nature of the risks will be examined.
b. *System description addressing risks* documents the scheme of the system intended to be built and its functions.
c. *Risk estimation* this phase consists of three sub-phases: analysing data, representing and modelling system related risks, and exploiting the model.

Prevention and mitigation choices: this phase depends on quantitative and qualitative risk estimation.

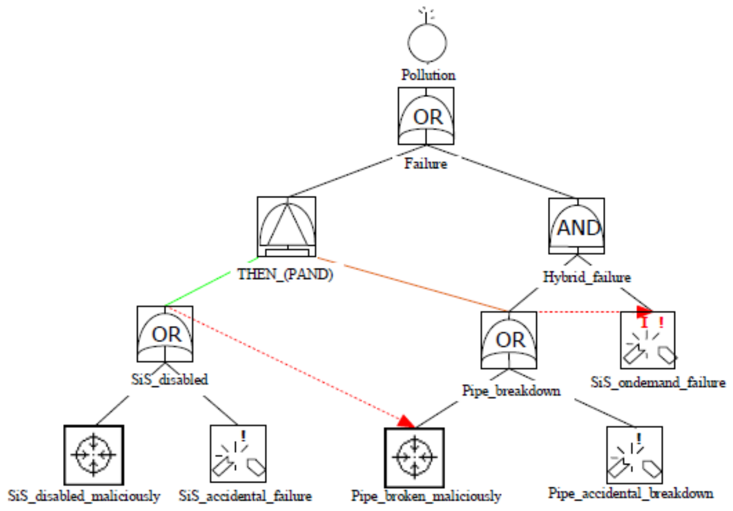

Figure 1 Modeling safety and security interdependencies with BDMP [21].

The newly founded technique was derived from a real case study [22] where the focus was on modeling case study about transporting a polluting substance with BDMP hoping towards more formal risk assessments.

3.3 KAOS for Safety

Knowledge Acquisition in automated Specification (KAOS) is a methodology for engineering requirements that enables analysts to build models requirements and to derive documents requirements from KAOS models (Figure 2). The meta-model for KAOS has been discussed in [23]. KAOS is a goal-oriented requirements engineering method intend to support the entire process of requirements analysis and elaboration – from high-level goals that need to be achieved to the requirements. Objects and operations notions assigned to various agents notion in the composite system provide a specification language, a tool support, and an elaboration method [24]. This section addresses KAOS for safety as well as artefact *Safety Obstacles* (Hazard).
The *Goal* model of KAOS looks like a tree that expresses relationships among goals of a system by showing how low-level goals contribute to higher-level goals and how in this goal model, an *AND-refinement* link relates a parent goal to a set of sub-goals that must be satisfied for the parent goal to be satisfied. Using KAOS goal refinement patterns are considered an efficient way to build the model because proofs can be reused. These patterns are capable of reducing time and cost of goal model construction.

An *Obstacle* meta-model is like a *Goal* meta-model notation (for security). However, the two are used to represent safety goals to reach obstacle treatment through the refinement into sub-obstacles. Each of these sub-obstacles is anchored with a new goal that works towards limiting and treating these obstacles. This method is implemented on the rest of the sub-obstacles until the goal '*Obstacle treatment*' is achieved, which is the main goal and is located on the top level of the KAOS diagram.

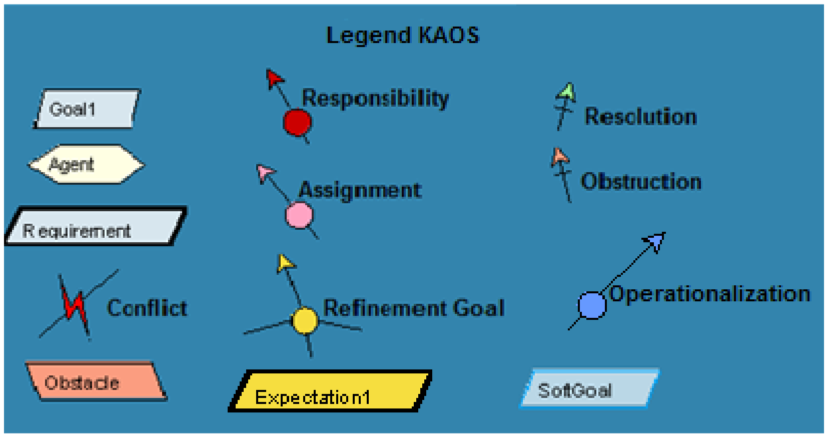

Figure 2. Legend KAOS

For safety requirements, it is very important to deal with *Obstacles* (hazard) KAOS element, which capture undesired properties. It allows analysts to identify and address exceptional circumstances during requirements engineering in order to produce robust or new requirements to avoid or reduce the impact of obstacles giving more reliable software [24].

The more specific the goal, the more specific its obstructing obstacles will be. As mentioned earlier, a high-level goal produces high-level obstacles that will be refined into much smaller sub-obstacles. These sub-obstacles are used for precise obstacle identification in order to evaluate their feasibility through agent behaviour negative scenarios. It is much easier and preferable to refine what is wanted than what is not wanted.

The level of how extensive obstacle identification is depends on the type and priority of the obstructed goal. For example, obstacle identification in *Safety Goals* needs to be adequately extensive. Domain-specific cost-benefit analysis needs to be performed to decide when the obstacle identification process should terminate.

Obstacle OR-refinement yields sufficient sub-obstacles to establish the obstacle; each OR-refinement of an obstacle obstructs the goal that is obstructed by this obstacle. Goals and AND/OR refinement of obstacles proceed exactly the same way except for only a few alternative OR refinements that are generally considered, in the case of obstacles, one may identify as many alternative obstacles as possible.

KAOS supports using semi-formal and linear formal specification language Linear Temporal Logic (LTL) to describe Goals, Obstacles and to perform logical proofs, which gives accuracy and reveals ambiguities. This is what sensitive and critical systems are in need for, which integrates between safety and security after identifying the requirements specifications of both and later reduced to formal languages that reveals complications resulted from achieving the goals of safety and security. Formal specifications can aid in correct design of system requirements specifications and improve the quality of system-to-be [25].

The semantic language of KAOS is necessary to ensure the correctness of the safety-critical requirements specifications described for developing the systems.

The following are definitions of elements found in KAOS:

1. *Goals* – descriptive milestones statements intended to be achieved.
2. *Agents* – active components like humans, devices, and legacy software that play a role towards achieving goals.

3. *Obstacle* – a condition if satisfied, may prevent a goal from being achieved and is used in producing an anti-model that shows why and by whom the original model can be threatened.
4. *Requirements* – a terminal goal that an agent is responsible for in the software to be developed.
5. *Object – any* entity defined in the system. An object has features and relations.
6. *Action* – the interaction between inputs and outputs within an object. Each action has pre-action, post-action, and triggers conditions.
7. *Operation model* – description of all behaviours whose requirements need to be fulfilled by agents. Behaviours are expressed in terms of operations that agents performed. Operations work on objects: they can create objects, trigger, state transitions of objects, and activate other operations [26].
8. *Responsibility model* – the responsibility model contains all responsibility diagrams. Each diagram describes the requirements and expectations an agent is responsible for, or has been assigned to them. An agent is assigned to expectations in a goal model [26].

3.4 KAOS for Security

This section will focus on the security extension of KAOS, hereafter referred to KAOS Security Extension (SE). The KAOS-SE is highly interrelated with the original KAOS methodology, and uses the model artifacts for security:

1- *Anti-goals* – attacker's own goals, including malicious obstacles to security goals.
2- *Attackers or attacker agents* – malicious agents in the environment.
3- *Anti-requirements* – terminal anti-goals that are realizable by the identified attacker agents.
4- *Vulnerabilities* – terminal anti-goals that are realizable by attackee software agents.
5- *Countermeasures* – a new security goal directed towards vulnerabilities or anti-requirements.

For security requirements analysis and elaboration by the use of *Anti-Goals* KAOS element, the goal notion allows the expression of security requirements patterns in terms of *anti-goals* notion and vulnerabilities of the system that is being studied. These patterns can also include a definition of the solution, or counter measure, to the attack in terms of goals that avoid a given vulnerability.

3.5 MUC - Misuse-Case

MUC diagrams, the conception of use cases is used to create and relate corresponding misuse cases used to address particularly security requirements [27]. The functionality of a system is modeled in use cases focusing on interactions with users and responses from the system. MUC extend the positive use cases with the negative ones to ensure eliciting security requirements.
A use case and a misuse case are related in using a directed association. If the association points line from a misuse case to a use case has the stereotype *'threaten'* while if the association points line from a security use case to a misuse case has the stereotype *'mitigate'*. It is stated that ordinary use cases represent requirements, security cases represent security requirements, and misuse cases represent security threats. The essence of the contained use cases is captured in an associated textual description since use case diagrams only give an overview of the system functionality.

Misuse cases are applicable to design a system that covers different security needs. It is possible to consider all three CIA (*Confidentiality, Integrity and availability*) goals. It incorporates common risk and threat analysis techniques.

The new elements that are introduced by misuse cases and related with the use case elements are [27]:

1- *Misuser* – an actor who initiates misuse cases, e.g., a hacker or thief.
2- *Misuse case* – actions that can cause harm to some system or stakeholder, e.g., stealing sensitive information.
3- *Threatens* – a relation where a use case is exploited by a misuse case, e.g., the registering of information in a system is exploited by stealing this information.
4- *Security use case* – a use case intended to mitigate misuse cases.
5- *Mitigates* – a relation where a security use case countermeasures a misuse case, e.g., protecting the sensitive information. There are three ways of describing misuse cases; a lightweight textual description, an extensive textual description and the misuse case diagram. These can either be used separately or in combination with each other. For both the textual descriptions there exist templates.

The process consists of five steps, which consists of (1) Identify critical assets in the system, (2) Define security goals for each asset, (3) Identify threats for each security goal, (4) Identify and analyze risks for the threats, (5) Define security requirements using mitigatation.

Misuse cases are applicable to design a system that covers different security needs. It is possible to consider all three CIA goals. It incorporates common risk and threat analysis techniques. Depending on the type of textual description used, it is in Sindre and Opdahl [27] described that the technique applies to security critical parts with the extensive description as well as the security related parts of a system through the lightweight description. It is stated that the technique has been used in E-shop and medical domains, along with a knowledge map and open web applications.

Several researches focused on narrowing the gap and comparisons between tools and techniques used in safety engineering and security engineering like Tor and Guttorm [28] where two approaches were compared against each other in using them to perform safety analysis relying on a use case. The comparison was between two methods MUC and FMEA (*Failure Mode and Effect Analysis*) and an experiment was conducted by two groups of students each on a method. The results conclude that MUC is better than FMEA in analysing failure modes related to user interaction and that MUC is easier and less confusing than FMEA. However, the results show that FMEA is better for in-depth analysis on failure modes related to the inner working of the system.

4. Accident Analysis Technique Languages

Accident techniques languages such as Swiss cheese model [29], AcciMap [30], STAMP [31] and STPA [32] classified as *systems-based accident analysis methods*. The approaches of these techniques are not domain-specific in *accident analysis* for a particular industry and what makes these approaches stand out is that the *socio-technical aspect* is taken into account during the analysis. Furthermore, these approaches are used in different industries such as aviation, defence, food, public health, oil and gas, and rail transport.

In Figure 3, shows the development of the tools over years and illustrates the classification of the tools under four categories. We selected the Swiss-Cheese model which takes in consideration human factor, the AcciMap which takes in consideration the organizational factor and the STAMP which takes in consideration system theory factor. Previously, we mentioned HAZOP which takes in consideration technical factor.

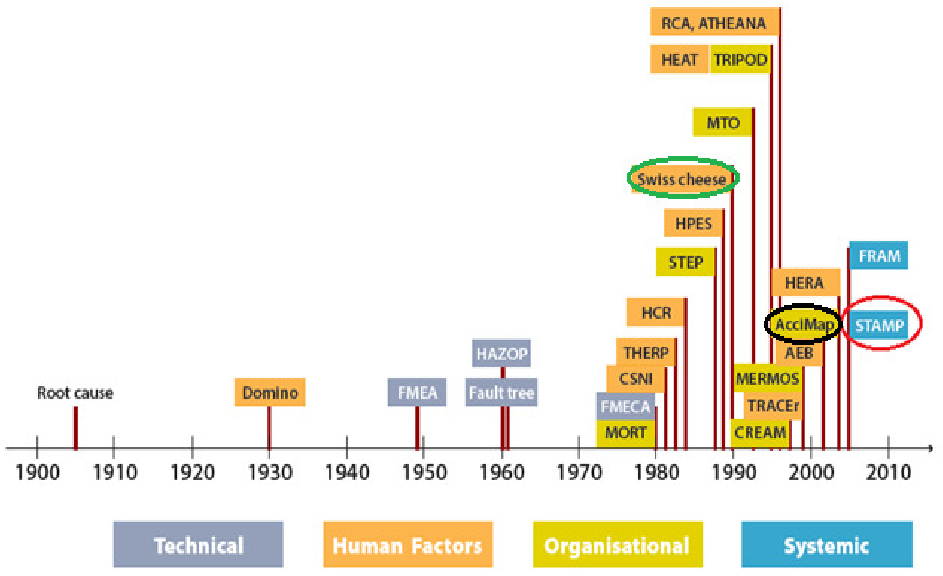

Figure 3. Accident Analysis and Risk Assessment Methods [33].

4.1. SCM - Swiss-Cheese Model

This model is considered one of the most famous accident causation models, also known as event-chain model. Reason [34] has famously developed a model based on the Swiss Cheese Metaphor (Figure 4) making use of slices and holes in cheese. The model suggests multiple contributors represented by the holes in cheese slices must be aligned for any adverse events to occur. Represented by the slices themselves, barriers in a system are intended to prevent errors that result in these unfavourable events. This SCM is not without drawbacks, and is not accepted uncritically [29].

Reason's model [34] describes the interaction between system wide '*latent conditions/potential conditions*' (a.k.a., inadequate designs and equipment, management and maintenance failures, lack in training and/or procedures) and unsafe acts made by human operators and their role in accidents. The model also describes the role of defences, such as protective equipment, rules and regulations, training, and engineered safety features, which are designed to prevent accidents. Weaknesses in these defences, created by latent conditions and unsafe acts [29], allow defences to be breached and accidents to occur.

Figure 4.1, Illustrates that, in the *Swiss-Cheese* model, an organization's defences against failure are modelled as a series of barriers, represented as slices of cheese. The holes in the slices represent weaknesses in individual parts of the system and are continually varying in terms of size and position across the slices. The system produces failures when a hole in each slice shortly aligns. If these holes aligned, it can allow the accident to occur [29].

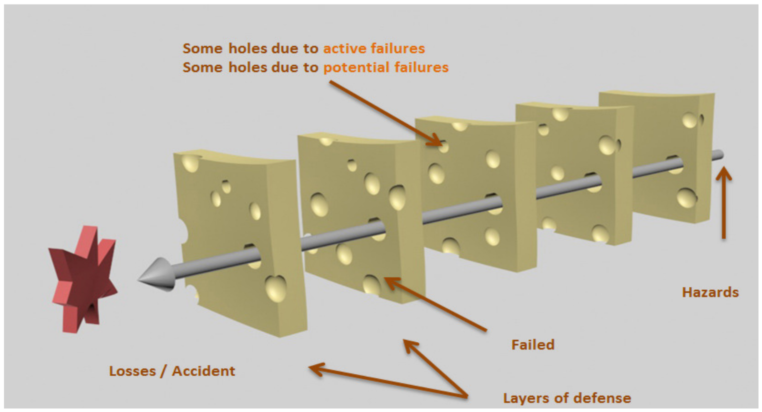
Some holes due to active failures
Some holes due to potential failures
Hazards
Failed
Losses / Accident
Layers of defense

Figure 4. Reason's Swiss cheese model, Adapted from [29].

4.2. AcciMap

The AcciMap accident analysis technique is based on Rasmussen's risk management framework. AcciMap is a generic approach and does not use taxonomies of failures across the different levels that are considered and designed specifically for analysing the causes of accidents and incidents that occur in complex socio-technical systems [35][36].

The AcciMap method involves the construction of a multi-layered causal diagram in which the various causes of an accident are arranged according to their causal remoteness from the outcome (depicted at the bottom of the diagram). The lower levels typically represent the immediate precursors to the event, relating to the activities of workers and to physical events, processes and conditions that contributed to the outcome. The next highest levels typically represent company and organisational-level factors. The highest levels generally incorporate governmental or societal-level causal factors, which are external to the organisation(s) involved in the event [30]. This way, the full range of factors that contributed to the event is modelled.

Figure 5, Illustrates that, Rasmussen's risk management framework [35] outlined the AcciMap method, which is used to graphically represent the system-wide failures, the precise format of the diagram varies depending on the purpose of analysis.

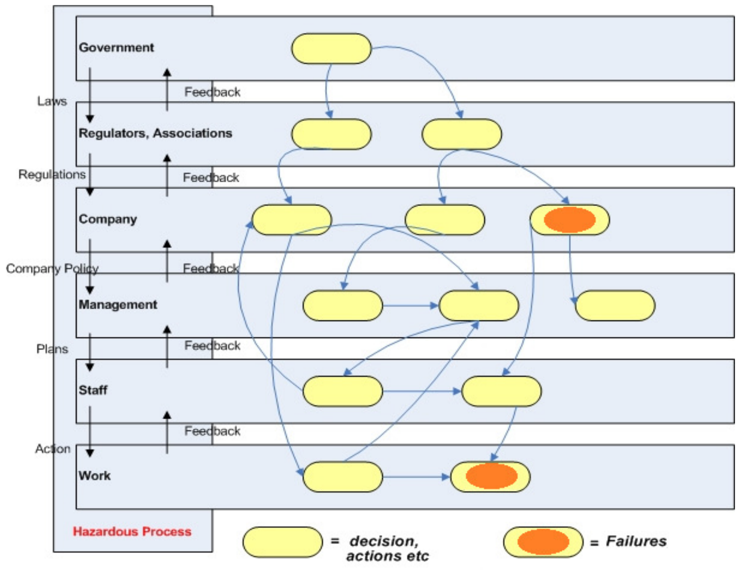

Figure 5. Rasmussen's risk management framework using AcciMap method, Adapted from [30].

4.3. STAMP - System-Theoretic Accident Model and Processes

STAMP was developed by Nancy G. Leveson [31], and is similar in theory with traditional hazard analysis methods (for example, Rasmussen's risk management framework, AcciMap). However, Nancy [37] considers these traditional hazard analysis methods only to deal with monitoring the accident flow and how it occurred. The traditional methods cannot explain how the accident happened in a component-nested system and taking the Socio-technical aspects into consideration, which in turn interacts with this system. It is considered as a cognitive hazard analysis method because it integrates all aspects of risk, including organizational and social aspects to understand accident causation.

Then, she published various publications extending the former, focussed on safety [38][39]. These publications are finally reinforced by some recent tutorials [40][41], presenting his work in-depth about safety engineering, which are part of resilience engineering [42].

STAMP [31] is the most recent approach to be developed, and considered a new accident causality model based on systems theory [43]. STAMP approach deals with safety through a technical language called Systems-Theoretic Process Analysis (STPA) that interacts with identifying hazards and hazard analysis; it can be used early in the system development life cycle to elicitation high level safety requirements and constraints in terms of identifying more causal factors and hazardous scenarios [43].

Young and Leveson [44][45] has developed an extension of Systems-Theoretic Process Analysis (STPA) to serve the hazard analysis for security engineering (Young used term *cyber-security*) and known as Systems-Theoretic Process Analysis for security (STPA-*Sec).* Young published some recent tutorials [32] presenting his work in-depth about security (*Cyber- Security*). Some tools that automate the activities of STPA were developed to support the hazard and accident analysis processes including A-STPA and SpecTRM.

5. Conclusions And Further Work

Integrating safety and security requirements should occur during the initial development phases of the system because it is a very important step for safety and security engineers in order to discover the causes of hazards and risks. Furthermore, it is vital since it's the only thing that covers the gap between safety engineers and security engineers especially since a security engineer knows the risks that a system could face and therefore has to protect the system and the equipment from any threats. On the other hand, a safety engineer does not know what the hazards would be or their effect on the environment. Therefore, a safety engineer will have to discover the unintentional hazard the system could possibly face. This is the critical point from which hazard and risks are derived by both types of engineers using an easy approach to communicate.

The assumption exists that the current security and safety requirements techniques and approaches, are only dedicated to their specific engineering domain. We surveyed the similar terminology, standards and modelling approaches that are established to elicit and validate them. Our remarks also indicate that the techniques and approaches for security and safety requirements cannot be simply interplayed but they require a systematic conceptual integration.

In the future work, we will develop a structured approach to integrate between security and safety by creating a SaS (Safety and Security) domain model. Furthermore, it will demonstrate that it is possible to use goal-oriented KAOS (Knowledge Acquisition in automated Specification) language in threat and hazard analysis to cover both safety and security domains making their outputs, or artifacts, well-structured and comprehensive, which will result in dependability due to the comprehensiveness of the analysis.

The developed structured approach will act as an interface for active interactions in risk and hazard management in terms of universal coverage, finding solutions for differences and contradictions which can be overcome by integrating the safety and security domains and using a unified system analysis technique (KAOS) that will result in analysis centrality.

REFERENCES

[1] Piètre-Cambacédès, L., & Chaudet, C. (2010). The SEMA referential framework: Avoiding ambiguities in the terms "security" and "safety". International Journal of Critical Infrastructure Protection, 3(2), 55-66. https://doi.org/10.1016/j.ijcip.2010.06.003

[2] Fabian, B., Gürses, S., Heisel, M., Santen, T., & Schmidt, H. (2010). A comparison of security requirements engineering methods. Requirements engineering, 15(1), 7-40. https://doi.org/10.1007/s00766-009-0092-x

[3] Eames, D. P., & Moffett, J. (1999, September). The integration of safety and security requirements. In International Conference on Computer Safety, Reliability, and Security (pp. 468-480). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-48249-0_40

[4] Firesmith, Donald G. Common Concepts Underlying Safety Security And Survivability Engineering. No. CMU/SEI-2003-TN-033. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2003.

[5] Firesmith, D. G. (2010, May). Engineering safety-and security-related requirements for software-intensive systems: tutorial summary. In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2 (pp. 489-490). ACM. https://doi.org/10.1145/1810295.1810444

[6] Allenby, K., & Kelly, T. (2001). Deriving Safety Requirements Using Scenarios. In Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on (pp. 228-235). IEEE. 10.1109/ISRE.2001.948563

[7] Srivatanakul, T., Clark, J. A., & Polack, F. (2004, September). Effective security requirements analysis: Hazop and use cases. In International Conference on Information Security (pp. 416-427). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-30144-8_35

[8] Alexander, Ian. Misuse Cases: Use Cases With Hostile Intent, Software, IEEE, vol.20, no.1, pp.58,66, Jan/Feb 2003. https://doi.org/10.1109/MS.2003.1159030

[9] Zafar, Saad, and R. Geoff Dromey. Integrating Safety And Security Requirements Into Design Of An Embedded System. 12th Asia-Pacific Software Engineering Conference (APSEC'05). IEEE, 2005. https://doi.org/10.1109/APSEC.2005.75

[10] Sun, M., Mohan, S., Sha, L., & Gunter, C. (2009, July). Addressing safety and security contradictions in cyber-physical systems. In Proceedings of the 1st Workshop on Future Directions in Cyber-Physical Systems Security (CPSSW'09).

[11] Sommerville, Ian. An Integrated Approach To Dependability Requirements Engineering. Current Issues in Safety-Critical Systems. Springer London, 2003. 3-15. https://doi.org/10.1007/978-1-4471-0653-1_1

[12] Stålhane, Tor, and Guttorm Sindre. Safety Hazard Identification By Misuse Cases: Experimental Comparison Of Text And Diagrams. International Conference on Model Driven Engineering Languages and Systems. Springer Berlin Heidelberg, 2008. https://doi.org/10.1007/978-3-540-87875-9_50

[13] Stålhane, Tor, Guttorm Sindre, and Lydie Du Bousquet. Comparing Safety Analysis Based On Sequence Diagrams And Textual Use Cases. International Conference on Advanced Information Systems Engineering. Springer Berlin Heidelberg, 2010. https://doi.org/10.1007/978-3-642-13094-6_14

[14] Piètre-Cambacédès, Ludovic, and Marc Bouissou. Cross-Fertilization Between Safety And Security Engineering. Reliability Engineering & System Safety 110 (2013): 110-126.https://doi.org/10.1016/j.ress.2012.09.011

[15] Kriaa, Siwar, Marc Bouissou, and Ludovic Piètre-Cambacédès. Modeling The Stuxnet Attack With BDMP: Towards More Formal Risk Assessments. 2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS). IEEE, 2012. https://doi.org/10.1109/crisis.2012.6378942

[16] Fovino, Igor Nai, and Marcelo Masera. Through The Description Of Attacks: A Multidimensional View. International Conference on Computer Safety, Reliability, and Security. Springer Berlin Heidelberg, 2006. https://doi.org/10.1007/11875567_2

[17] Raspotnig, Christian, and Andreas Opdahl. Comparing Risk Identification Techniques For Safety And Security Requirements. Journal of Systems and Software 86.4 (2013): 1124-1151. https://doi.org/10.1016/j.jss.2012.12.002

[18] Winther, R., Johnsen, O. A., & Gran, B. A. (2001, September). Security assessments of safety critical systems using HAZOPs. In International Conference on Computer Safety, Reliability, and Security (pp. 14-24). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45416-0_2

[19] Ericson, C. A. (2015). Hazard analysis techniques for system safety. John Wiley & Sons.

[20] Hansen, K. M., Wells, L., & Maier, T. (2004). HAZOP analysis of UML-based software architecture descriptions of safety-critical systems. Proceedings of NWUML, 59-78.

[21] Piètre-Cambacédès, L., & Bouissou, M. (2010, October). Modeling safety and security interdependencies with BDMP (Boolean logic Driven Markov Processes). In Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on (pp. 2852-2861). IEEE.

[22] Kriaa, S., Bouissou, M., & Piètre-Cambacédès, L. (2012, October). Modeling the Stuxnet attack with BDMP: Towards more formal risk assessments. In 2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS) (pp. 1-8). IEEE.

[23] Matulevičius, R., Heymans, P., & Sindre, G. (2015). COMPARING GOAL-MODELLING TOOLS WITH THE RE-TOOL EVALUATION APPROACH*. Information Technology And Control, 35(3).

[24] Van Lamsweerde, A., & Letier, E. (2000). Handling obstacles in goal-oriented requirements engineering. IEEE Transactions on Software Engineering, 26(10), 978-1005. https://doi.org/10.1109/32.879820

[25] Nakagawa, H., Taguchi, K., & Honiden, S. (2007, November). Formal specification generator for KAOS: model transformation approach to generate formal specifications from KAOS requirements models. In Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering (pp. 531-532). ACM. https://doi.org/10.1145/1321631.1321729

[26] Respect, I. T. (2007). A KAOS Tutorial.

[27] Sindre, G., & Opdahl, A. L. (2005). Eliciting security requirements with misuse cases. Requirements engineering, 10(1), 34-44. https://doi.org/10.1007/s00766-004-0194-4

[28] Stålhane, T., & Sindre, G. (2007, November). A comparison of two approaches to safety analysis based on use cases. In International Conference on Conceptual Modeling (pp. 423-437). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-75563-0_29

[29] Perneger, T. V. (2005). The Swiss cheese model of safety incidents: are there holes in the metaphor?. BMC health services research, 5(1), 1. https://doi.org/10.1186/1472-6963-5-71 PMid:16280077 PMCid:PMC1298298

[30] Salmon, P. M., Cornelissen, M., & Trotter, M. J. (2012). Systems-based accident analysis methods: A comparison of Accimap, HFACS, and STAMP. Safety science, 50(4), 1158-1170. https://doi.org/10.1016/j.ssci.2011.11.009

[31] Leveson, N. (2004). A new accident model for engineering safer systems. Safety science, 42(4), 237-270. https://doi.org/10.1016/S0925-7535(03)00047-X

[32] Young Jr, W. E. (2014). STPA-SEC for cyber security mission assurance. Eng Syst. Div. Syst. Eng. Res. Lab.

[33] EUROCONTROL, A. (2009). White Paper on Resilience Engineering for ATM. Report of the Project Resilience Engineering for ATM.

[34] Reason, J. (1990). The contribution of latent human failures to the breakdown of complex systems. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 327(1241), 475-484. https://doi.org/10.1098/rstb.1990.0090 PMid:1970893

[35] Rasmussen, Jens. "Risk management in a dynamic society: a modelling problem." Safety science 27.2 (1997): 183-213. https://doi.org/10.1016/S0925-7535(97)00052-0

[36] Rasmussen, J., & Suedung, I. (2000). Proactive risk management in a dynamic society. Swedish Rescue Services Agency.

[37] Leveson, N. G. (2004). A systems-theoretic approach to safety in software-intensive systems. IEEE Transactions on Dependable and Secure computing, 1(1), 66-86. https://doi.org/10.1109/TDSC.2004.1

[38] Nancy G. Leveson, "An STPA Primer Version 1", August 2013b.

[39] Nancy G. Leveson, "Tutorial: Engineering a Safer World", STAMP Conference, 2012.

[40] Nancy G. Leveson, "Tutorial: Engineering a Safer World", STAMP Conference, 2013.

[41] Nancy G. Leveson, "Tutorial: Engineering a Safer World", STAMP Conference, 2014.

[42] Woods, D. D., Leveson, N., & Hollnagel, E. (Eds.). (2012). Resilience engineering: concepts and precepts. Ashgate Publishing, Ltd..

[43] Leveson, N. (2011). Engineering a safer world: Systems thinking applied to safety. Mit Press. PMid:21478425

[44] Young, W., & Leveson, N. (2013, December). Systems thinking for safety and security. In Proceedings of the 29th Annual Computer Security Applications Conference (pp. 1-8). ACM. https://doi.org/10.1145/2523649.2530277

[45] Young, W., & Leveson, N. G. (2014). An integrated approach to safety and security based on systems theory. Communications of the ACM, 57(2), 31-35. https://doi.org/10.1145/2556938