Security-Informed Safety Case Approach to Analysing MILS Systems

Kateryna Netkachova
City University London and Adelard LLP
London, UK
Kateryna.Netkachova.2@city.ac.uk

Kevin Müller
Airbus Group Innovations
Munich, GERMANY
Kevin.Mueller@airbus.com

Michael Paulitsch
Thales Austria GmbH
Vienna, AUSTRIA
Michael.Paulitsch@thalesgroup.com

Robin Bloomfield
City University London and Adelard LLP
London, UK
reb@adelard.com

ABSTRACT
Safety cases are the development foundation for safety-critical systems and are often quite complex to understand depending on the size of the system and operational conditions. The recent advent of security aspects complicates the issues further. This paper describes an approach to analysing safety and security in a structured way and creating security-informed safety cases that provide justification of safety taking into particular consideration the impact of security. The paper includes an overview of the structured assurance case concept, a security-informed safety methodology and a layered approach to constructing cases. The approach is applied to a Security Gateway that is used to control data flow between security domains in a separation kernel based operating system in avionics environment. We show that a clear and structured way of presenting a safety case combining safety and security alleviates understanding important interactions taking into account the impact and, hence, increases safety.

General Terms
Design, Security, Theory.

Keywords
Security-informed safety case, MILS, security gateway, layers of assurance.

1. INTRODUCTION
Traditionally, safety and security have been treated as separate disciplines for high assurance Cyber-Physical Systems. This position is becoming untenable as there is a growing realization that security and safety are closely interconnected especially with the increasing openness of systems. Both security and safety can be viewed as aspects of dependability in the sense that each is concerned with mitigating the effects of a particular kind of failure. Both disciplines use similar techniques to identify potential failure modes or attack vectors and assess their impact on the overall system. Thus, there is considerable overlap between safety and security, and it is often important to address them together.

A detailed safety analysis and the production of a safety case are now required by various standards in automotive [1], railway [2], nuclear [3] and other industries. In this paper we are focusing on the avionics domain, but leverage the formality of safety cases of other domains. Aviation has been implementing systems by following the safety standards [4] [5] and recently by using the safety-driven design-approach of Integrated Modular Avionics (IMA) [6]. IMA relies on the concept of partitioning between applications. Similarly, avionic industry has recently issued security standard [7] [8]. In terms of the system design, the security-driven design approach of Multiple Independent Level of Security (MILS) is used with partitioning and a separation kernel to control information flow [9]. Building on the safety case approach and taking into account security considerations, we are interested in developing an integrated security and safety solution that can be applied to the avionics use case to create a good and convincing security-informed safety case. As a use case, a MILS-based gateway controlling information flow between aircraft security domains, is presented and analysed.

The paper is structured in the following way. The overall approach that we are developing is described in Section 2. An overview of the case study and the application of the approach are discussed in Section 3. Conclusions and next steps are outlined in Section 4.

2. SECURITY-INFORMED SAFETY CASE

2.1. General Concept
The approach we are developing is based on the use of structured assurance cases for communicating and building confidence in the safety and security properties of the system. Structured assurance cases are used in a wide range of industrial domains. Our current practice is based on a concept of Claims, Arguments and Evidence (CAE), which can be related to the approach developed by Toulmin [10]. CAE supports the description of how sophisticated engineering arguments are actually made. The key elements of CAE are briefly described in Section 6.1 and also [11].

Security considerations have a significant impact on various aspects of safety justification. It is necessary to make claims about security properties as well as safety properties, to demonstrate compliance with both security and safety standards, and to consider a broader set of potential hazards, threats and vulnerabilities.

Our experience and previous research [12] has shown that a significant portion of a security-informed safety case will need to address security explicitly. In some instances this will lead to substantial changes to the system design, the implementation process and the justification. For example, the following areas are particularly significant from a security perspective and need more scrutiny in a security-informed safety case:

- Supply chain integrity
- Confidentiality of the process and product
- Combination of specific aspects of addressing safety and security design approaches and life cycle processes of certification and updates, such as stable and infrequently changing system design and certification approaches addressing safety combined with requirement of frequent security subsystem updates.

- Issues of lifecycle threats and malicious threats to evidence, e.g.:
    - Malicious events after system deployment that will change in nature and scope as the threat environment changes
    - Weakening of security controls as the capability of the attacker and technology changes. This may have major impact on proposed lifetime of installed equipment and design for refurbishment and change
- Design changes to address new user interactions, training, configuration, and vulnerabilities. This might lead to additional functional requirements that implement security controls.
- Possible exploitation of the device/service to attack itself or others.

In order to address these additional security risks within a case, it is necessary to combine safety and security risk assessment. Because many systems already have safety justifications with corresponding risk assessments we are developing an adapted process to make them security-informed. Thus, our approach is different from other work in avionics, for example, where the idea is to develop an integrated approach from scratch. Furthermore, the presented approach brings the formality of safety case perspective into avionics, where the term safety case is not common at the moment.

2.2. Security-Informed Safety Case Architecture

The justification of a security-informed safety case can be complex, or at least complicated, as it combines the claims from adaptation, supply chain and deployment, implementation details and hazard and vulnerability analysis. As one role of the case is to communicate effectively, one needs to balance both the risk of abstracting away important details and the risk of the important details being lost in a sea of other details.

The works of Rushby and Delong [13] [14] raise the idea of compositionality and layered assurance. The goals of the approach are manifest in the LAW series of workshop [15]. These explore the "bold proposition that it is possible to build assured systems from compositions of previously assured components, while being able to derive the system level properties (e.g., safety & security) systematically from the properties of the components. LAW spans the theoretical, engineering, and certification challenges to be met in making compositional assurance for such systems a reality. They use the term "layered" assurance to encompass diverse manifestations of combined assurance, including composition (of assured components), incremental certification (incremental cost for incremental change), abstraction layers (building upon assurance of lower layers), and polymorphism (common assurance of variants, such as among members of a product family). MILS is one approach to achieving the goals of compositional assurance.

Abstraction is one of the key structuring mechanisms and we have experimented with various levels of abstraction when creating security-informed safety cases. We call those levels *layers of assurance*, because within each abstraction level the assurance is provided. We identified the following main layers of assurance:

- L0 Policy and requirements – the highest level of abstraction where the system represents its requirements, and defines safety and security policies and their interaction;
- L1 Architectural layer – the intermediate level where the abstract system components and architecture are analysed;
- L2 Implementation layer – the detailed level where the implementation of specific components and their integration within the specific system architecture are scrutinised.

These layers of assurance fit well the layered system design approach of aerospace described in ARP 4754 [6] combined with the compositional approach of MILS [10] and IMA [16].

Our approach supports the goals of layered and compositional assurance. Additionally, it also shows where the ambition needs to be modified. We take into account the diversity of properties. Exemplarily, properties can be:

- properties of the system not the components
- of wider scope at the architectural level (i.e. they can inform the L0 level but the overall assurance may need additional considerations)
- derived properties that are identified only when the implementation details are defined and analysed (e.g. related to derived hazards, and so forth.

The following chapter instantiates and discusses each abstraction layer in application to the MILS gateway use case, a potential subsystem in future aircraft system architectures.

3. ANALYSIS OF SECURITY GATEWAY

3.1. Overview of the Gateway Use Case

As use case for our study we use a gateway function. Usually security gateways connect two or more security domains [17] to each other and control the information flow according to a given information flow policy. Our gateway is implemented by using the design concept of Multiple Independent Layers of Security (MILS). MILS bases on the properties of separation and controlled information flow. In current MILS proposals these two properties are achieved by a special operation system layers. This operating system provides separation by the concept of partitioning. Partitions are isolated runtime environments for applications. Since a meaningful MILS system only works with allowed interactions among application, the operating system also allows a controlled information flow between partitions. Applications can implement further layers of security, e.g. cryptographic algorithms or more sophisticated data processing, or entire other function such as image processing.

The gateway is intended to filter application-level data traffic for the TFTP and HTTP protocol. It achieves its data filtering by a cooperating of several partition applications allowing to interact in a certain way with each other. Figure 1 shows the abstract system design [18]. Each solid box represents one partition provided by the operating system. In the use case implementation we use PikeOS [19]. The arrows define the directed and controlled information flow among partition in order to allow interaction. Using the MILS architecture allows the definition and implementation of a local security and information flow policy for each partition. For example, each Receiver Component is in charge to analyse ingress traffic and forward it either to the TFTP or to the HTTP filter chain.

Each filter chain's policy again is to filter these data packets according to the prevalent application-level protocol.

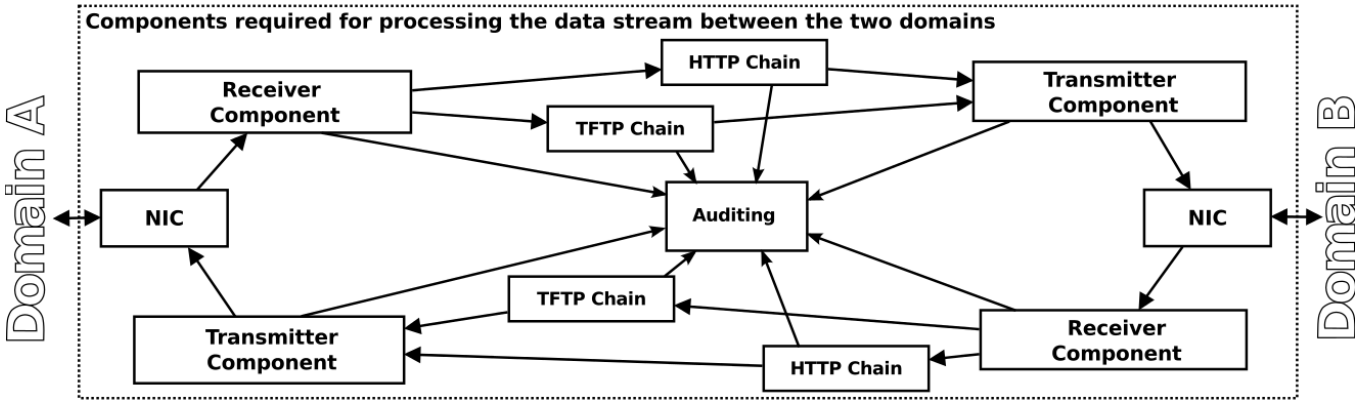

Figure 1: High-level View of Gateway Components

The security policy of the Audit partition is to gather audit records generated by other partitions and to store them securely in order to provide traceability.

3.2. Application of the Layered Assurance Approach

3.2.1. Scope of the System

The scope of the security and safety analysis, of our use case is broader than just the gateway. We are interested in a system consisting of applications in two domains having different security levels. Each application belongs to a single security domain and can only communicate with applications from the other security domain via the security gateway.

Figure 2 shows a context data flow diagram representing the high view of such broader system. The double circle at the centre represents the gateway process that performs various operations (receives data streams, applies security policies, transmits the filtered content etc.). Rectangles identify external entities that interact with the gateway. They comprise applications from two domains (A and B) and the administrators that can maintain the gateway while being in maintenance mode (a special operational mode). An open-ended rectangle indicates a data store where the logging and alerting data are stored by the gateway for later use.

3.2.2. Discussion of L0 and the Case Study

The first and most abstract level concerns requirements, policies and principles of the system, with the focus on the system safety, system security and their interaction.

The top-level claim involves introducing a security policy, considering a set of applications at different security classification, and safety criticalities associated with them. At L0 the abstract gateway enforces a security policy that puts constraints on inter-domain information flows. We need to undertake an analysis to show that the interaction and trade-offs are satisfactory. It is unlikely that under all runtime circumstances one simple and static policy will be valid. There will be times of initialisation, special operational modes or changing threat levels that will impact the policy. For example at high levels of security threat the system might be adjusted to isolate high-safety applications from all applications of other domains. Alternatively, in times of operationally challenging conditions safety consideration could require an adaption of the security policies in order to allow manually transmitted messages by trusted external entities to provide guidance and recovery strategies to the pilots.

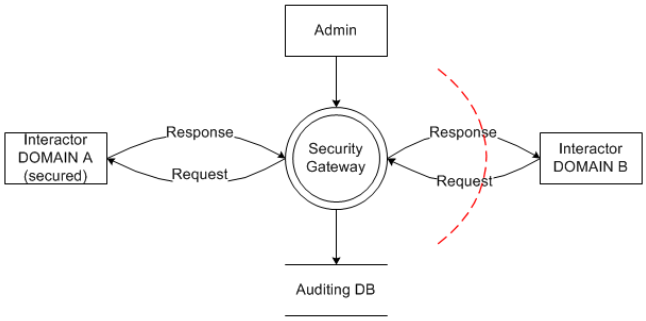

Figure 2: The high-level data flow diagram of the system

Figure 3 provides an illustration of some of these considerations by showing different policy zones. At the bottom left we have an area of maximum operational benefit. The other areas indicate how certain threat level security concerns dominating e.g. the need to restrict the flows. In this case the safety analysis must show that these are acceptably safe even if they do cause higher workload or operational complexities. There is a corresponding zone where safety issues dominate and the security policy is the same or weakened. In this case, the security analysis must show that identified security threats are countered by other environmental properties during this time.

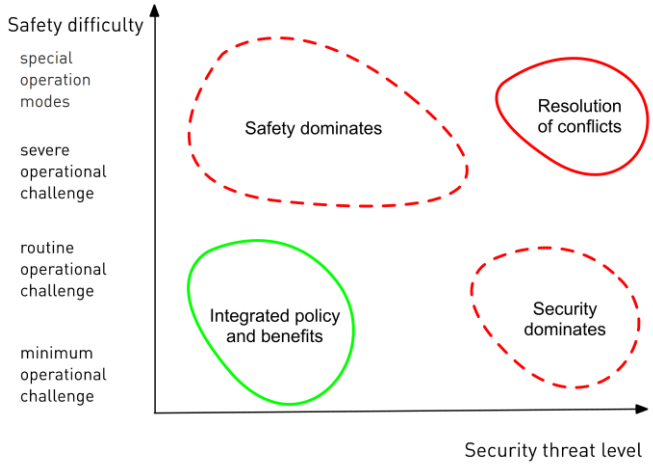

Figure 3: Defining integrated policy

Finally, the top right hand corner is a very uncertain and undecidable area where some special capabilities might be needed, e.g. in the form of a manual override to security policy enabling flows as well as a manual closing down on all non-essential information flows if threats were high and compromise was likely. Again, the consequences of any trade-offs need to be assessed during the analysis.In summary as we create the L0 case for the system we need to address the:

- modes of safety application
- operational modes of the gateway
- impact of different threat levels
- attributes for the gateway's policy

At this level we develop a substance to the analysis of the policy interactions, we have an updated security policy and safety requirements as well as initial results from the risk analysis. Also, we identify some more details about modes of operation of the gateway and the overall system as well as availability and other attributes for the properties.

3.2.3. Discussion of L1 and the Case Study

At this level we analyse the components and the architecture of the system, which play important roles in achieving system objectives and enforcing the critical properties of the system. To address security considerations, we applied various methods of security analysis including:

- A guideword-based approach derived from the safety HAZOP analysis;
- An analysis of trust relationships;
- STRIDE, the Microsoft threat modelling approach.

The analysis conducted showed that many parts of the critical properties of the system are enforced by the separation kernel. Other important components of the architecture include the gateway application software, system monitor & audit component, system integrator, etc.

The resulting L1 level case considers many of the critical properties, related to the system functionality as well as other aspects (reliability, availability, etc.) and properties identified at L0. Each of the architectural components enforcing the critical properties of the system should be expanded further to demonstrate that the implemented subcomponents really enforce the corresponding properties. This type of analysis is to be performed at the next L2 level of abstraction.

3.2.4. Discussion of L2 and the Case Study

L2 represents the detailed implementation level. At this level we introduce all the technical information available about the actual system implementation.

To illustrate the approach let us consider one of the critical properties of the system identified at L0: "All communication between domains of different security levels must be controlled in accordance with a system-wide security policy". At the L1 level this is expanded into subclaims related to system components and architecture. One of such subclaims defined at L1 is: "All communication between domains of different security levels must go through the security gateway". The L1 analysis also shows that this subclaim is enforced by the defined information flow between partitions configured and assured by the PikeOS component within the system architecture. Therefore, the implementation details of both the gateway and the PikeOS need to be thoroughly analysed in order to demonstrate that the critical property is really enforced. This analysis is performed at the L2 level of abstraction.

In terms of implementation, the security gateway is composed out of user applications hosted by PikeOS. The gateway's purpose is to control all information flow between applications located in different security domains according to the system-wide policy. All applications are supposed to communicate in accordance to the settings specified in the PikeOS system configuration file (VMIT file). This file is configured in an XML format by the system integrator and converted into a binary form. Then all software application binaries (with binaries of the security gateway component being part of them), PikeOS binary objects (microkernel, platform support package, PikeOS system software) and the PikeOS system configuration file are assembled into one binary file - PikeOS ROM image, which is then booted and run on the target system [20].

Therefore, the case created at this level of abstraction may include (but is not limited to) the following claims:

1) Gateway is implemented and deployed as a user applications within PikeOS partitions;
2) Inter-partition channels in the VMIT file are configured in a way that all partitions with different security settings can only communicate with the gateway partitions (inbound or outbound);
3) There are no errors in the configuration file allowing partitions with different security settings to communicate bypassing the gateway partitions;
4) Partitions can only communicate by using the communication channels provided by PikeOS (e.g. via shared memory resources, network resource, etc.)
5) All PikeOS binary objects, configuration binary, gateway application binary and the resulting ROM binary image are generated properly without any malicious modifications or corruptions;
6) The binary objects are not modified or replaced after they have been generated.
7) All partitions are initialized, created and set up correctly before data passes the gateway;
8) Gateway application is loaded properly after the separation kernel has been established, and is available to use;
9) Gateway application can receive and send messages to the gateway partitions' ports, which are the end points for the communication channels configured in the VMIT file;
10) PikeOS security kernel correctly enforces any settings specified in the VMIT configuration file.

Some of the above details (e.g. points #4, 6, 7, 10) are related to the general PikeOS implementation and can be satisfied by showing that the PikeOS is implemented correctly to its specification providing all the required functionality. Other claims are dependent on the correct implementation of the gateway application (#1, 9), experience of the system integrator (#2, 3), trusted development and deployment processes (#5, 6).

As a result, we developed CAE structures for each of the levels when analysing the gateway. There are two main roles of the approach: 1) it aids in the communication by providing a summary of the issues and their interrelationship and 2) it indicates how we might reason that the lower properties combine to satisfy the top-level claim. To show that the claims are a complete set and that the PikeOS and Security Gateway properties do in fact combine in this way will require us to provide a more formal semantics. One way to do this is to take a more explicit model based approach where the claim structure helps us identify the right level of abstraction and detail in the model. This is a topic for future research.

4. CONCLUSION

This paper presents the approach to creating security-informed safety cases and its application to a MILS-based security gateway. We have developed a Layered Assurance approach and provided some findings from the case study.

We found that the Layered CAE Assurance Case has broadened our view of how to combine safety and security. For example since tackling at architecture level is insufficient, we need to escalate to requirements using the abstraction layers and the explicit consideration of policy interactions within the L0 layer. In addition the consideration of lifecycle issues, particularly the adaptation and updating of the system that is part of our assurance case approach.

The CAE Layered Approach provides a generic link between a number of key processes: the integrated risk analysis and the safety and security system development lifecycle, and further integration could be developed.

We also found that the IMA architecture and PikeOS have intrinsic properties of separation and partitioning that are fundamental to enforcing the safety and security properties. While many of the assurance required from a security-informed safety perspective will exist in the IMA assurance, the emphasis on the credibility of the supply chain, the trust in tools, the response to malicious events, maintenance and update policies will be different.

The next steps will be taken towards additional formalisation of the reasoning within the security-informed safety cases and linking them to formal models. Additionally, issues related to compositionality and traceability between layers would need to be addressed in more detail.

5. ACKNOWLEDGMENTS

This work was supported by the Artemis JU SESAMO project (project number 295354), the European Union's 7th Framework Programme project EURO-MILS (ID: ICT-318353) and the UK EPSRC funded Communicating and Evaluating Cyber Risk and Dependencies (CEDRICS) project which is part of the UK Research Institute in Trustworthy Industrial Control Systems (RITICS).

6. APPENDIX

6.1. Assurance Case Overview

Structured assurance cases are used in a wide range of industrial domains. Current practice is based on a concept of claims, arguments and evidence (CAE), which can be traced back to the approach developed by Toulmin [10]. CAE supports the description of how sophisticated engineering arguments are actually made. The key elements are:

- Claims, which are assertions put forward for general acceptance. They are typically statements about a property of the system or some subsystem. Claims that are asserted as true without justification become assumptions and claims supporting an argument are called sub-claims.
- Evidence that is used as the basis of the justification of the claim. This is information accepted as an established fact. Sources of evidence may include the design, the development process, prior field experience, testing (including statistical testing), source code analysis or formal analysis.
- Arguments link the evidence to the claim. They are the "statements indicating the general ways of arguing being applied in a particular case and implicitly relied on and whose trustworthiness is well established" [10] together with the validation for the scientific and engineering laws used.

The basic elements of a justification represented in a graphical ASCAD notation [11] shown in Figure 4.

The above structure can be extended by expanding the sub-claims to further sub-claims, with the arguments justifying that the decompositions into sub-claims are valid. The final nodes of the justification should always be supporting evidence.

The motivation for an assurance case is to:

- provide an assurance viewpoint - for efficient review;
- provide a focus and rationale for assurance activities - leading to efficient analysis and evaluation;
- provide a reviewable approach - so that all stakeholders can be involved;
- demonstrate the discharge of duty to public and shareholders;
- allow interworking between standards and innovation.

There is considerable standardisation work on structured cases and Claims Argument Evidence and activities internationally in a number of sectors. Of particular relevance is the ISO/IEC standard that provides a definition of the CAE concept [21].

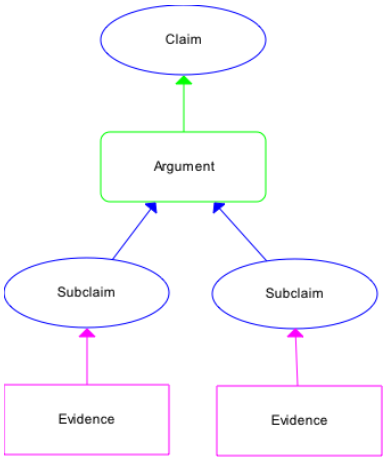

Figure 4: Example of CAE structure in an assurance case

7. REFERENCES

[1] International Organization for Standardization, "ISO 26262: Road Vehicles - Functional Safety".

[2] CSN EN 50129, "Railway applications - Communication, signalling and processing systems - Safety-related electronic systems for signalling," 2003.

[3] Ministry of Defence, "Def Stan 00-55: Requirements for safety related software in defence equipment".

[4] EUROCAE/RTCA, "ED-135/ARP4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," 1996.

[5] EUROCAE/RCTA, "ED-79A/ARP4754A: Guidelines for Development of Civil Aircraft and Systems," 2010.

[6] Airlines Electronic Engineering Committee, "ARINC 653: Avionics Application Software Standard Interface," 2010.

[7] EUROCAE/RCTA, "ED-202 / DO-326: Airworthiness Security Process Specification," 2010.

[8] EUROCAE, "ED-203: Airworthiness Security Methods and Considerations," 2012.

[9] C. Boettcher, R. Delong, J. Rushby and S. Wilmar, "The MILS Component Integration Approach to Secure Information Sharing," in *27th Digital Avionics Systems Conference (DASC)*, St. Paul, MN, 2008.

[10] S. Toulmin, "The Uses of Argument," Cambridge University Press, 1958.

[11] Adelard, "ASCAD - Adelard Safety Case Development Manual," 2010.

[12] R. Bloomfield, K. Netkachova and R. Stroud, "Security-Informed Safety: If it's not secure, it's not safe," in *5th International Workshop on Software Engineering for Resilient Systems (SERENE)*, Kiev, Ukraine, 2013.

[13] R. Delong, "Compositional Certification Lecture Notes," in *Real-Time Embedded Systems Forum, The Open Group (TOG) conference*, Toronto, Canada, 2009.

[14] C. Boettche, R. Delong, J. Rushby and W. Sifre, "The MILS Component Integration Approach to Secure Information Sharing," in *27th IEEE/AIAA Digital Avionics Systems Conference (DASC)*, St. Paul MN, 2008.

[15] ASCAS, "The Layered Assurance Workshop (LAW)," 2007-2013.

[16] RTCA, "DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations," 2005.

[17] Airlines Electronic Engineering Committee, "ARINC 811: Commercial Aircraft Information Security Concepts of Operation and Process Framework," 2005.

[18] E.-M. project, "D11.1 - Project Requirements: Classification, Cross-domain analysis and High-Level Architecture," 2014.

[19] SYSGO AG, "PikeOS Manual - v3.4," 2014.

[20] SYSGO AG, "Installing and Using PikeOS," 2014.

[21] ISO/IEC 15026-2, "Systems and software engineering -- Systems and software assurance -- Part 2: Assurance case," 2011.