D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

Version 01

2 May 2013

Final

Dissemination level: PU

Contributing partners
ADEL, CITY, CTU, DICEA, DTU, EDF, ESY, FTW, IFAG, IIT-CNR, SAG, UC
TABLE OF CONTENTS

1 Introduction ........................................................................................................................................... 9
  1.1 Document structure .......................................................................................................................... 9
  1.2 Technical context and objectives .................................................................................................... 10

2 Stochastic models of interdependent infrastructures (CITY) ............................................................... 12
  2.1 Motivation ......................................................................................................................................... 12
  2.2 Overview .......................................................................................................................................... 13
    2.2.1 Safety and Security metrics ........................................................................................................ 15
    2.2.2 Overview of the technical paper “Preliminary Interdependency Analysis (PIA): Method and tool support” ............................................................................................................................................. 16
  2.3 Application to the SESAMO use cases ............................................................................................... 18
    2.3.1 AKHELA .................................................................................................................................... 18
  2.4 Possible applications to other SESAMO use cases ........................................................................... 18
    2.4.1 EDF ......................................................................................................................................... 18
  2.5 Appendix .......................................................................................................................................... 18

3 Security-informed Safety Cases (CITY and ADEL) .............................................................................. 19
  3.1 Motivation ......................................................................................................................................... 19
  3.2 Scientific Contribution ....................................................................................................................... 19
  3.3 Application to the SESAMO use cases ............................................................................................... 23

4 Message authentication and schedulability analysis on CAN bus (CTU) .......................................... 25
  4.1 Motivation ......................................................................................................................................... 25
  4.2 Scientific Contribution ....................................................................................................................... 26
  4.3 Application to the SESAMO use cases ............................................................................................... 26

5 Pareto frontier between safety and security constraints (DTU) ........................................................... 27
  5.1 Motivation ......................................................................................................................................... 27
  5.2 Scientific Contribution ....................................................................................................................... 27
  5.3 Application to the SESAMO use cases ............................................................................................... 28

6 Safety and Security in the presence of Denial of Service Attacks (DTU) ............................................ 30
  6.1 Motivation ......................................................................................................................................... 30
  6.2 Scientific Contribution ....................................................................................................................... 31
    6.2.1 Overview “Safety versus Security in the Quality Calculus” .......................................................... 32
  6.3 Application to the SESAMO use cases ............................................................................................... 32

7 The KB3 workbench ............................................................................................................................... 35
  7.1 Motivation ......................................................................................................................................... 35
  7.2 Scientific Contribution ....................................................................................................................... 35
  7.3 Application to the SESAMO use cases ............................................................................................... 36
  7.4 References ....................................................................................................................................... 38

8 Safety & security analysis of resilient services in communication networks (FTW) 39
  8.1 Motivation ......................................................................................................................................... 39
  8.2 Scientific Contribution ....................................................................................................................... 39
  8.3 Application to the SESAMO use cases ............................................................................................... 40
    8.3.1 Overview of the technical paper “Timed Broadcast via Off-the-Shelf WLAN Distributed Coordination Function for Safety-Critical Systems” included in the Appendix ......................................................... 41

9 Formal Metrics (IIT-CNR) ................................................................................................................... 42
  9.1 Motivation ......................................................................................................................................... 42
  9.2 Scientific contribution ....................................................................................................................... 42
D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

9.2.1 Analysis ............................................................................................................................................. 43
9.3 Assessment of a system with an adaptive attacker ........................................................................... 43
9.3.1 Motivation ........................................................................................................................................... 43
9.3.2 Scientific contribution .......................................................................................................................... 44
9.3.3 Attacker’s behaviour ............................................................................................................................ 45
9.4 Application to the Medical Use Case .................................................................................................. 46
9.5 Future work ........................................................................................................................................... 46
9.6 References: ........................................................................................................................................... 47

10 State of the art FMEA techniques (SAG) ............................................................................................... 48
10.1 FMEA overview and fundamentals .................................................................................................. 48
10.2 History and Relation to standards .................................................................................................... 49
10.3 Classification and Flavors ................................................................................................................... 49
10.4 Failure Mode, Effects and Criticality Analysis .................................................................................. 51
10.5 General FMEA Approach .................................................................................................................. 51
10.6 The Safe Failure Fraction SFF .......................................................................................................... 56
10.7 Calculation of PFD (Probability of failure on demand) ....................................................................... 61
10.8 References .......................................................................................................................................... 62

11 Safety and security analysis for railway application (UC) ................................................................. 64
11.1 Motivation ............................................................................................................................................ 64
11.2 Scientific Contribution ........................................................................................................................ 67
11.2.1 Safety analysis and evaluation in railway industry ......................................................................... 67
11.2.2 Security analysis and evaluation in railway industry ..................................................................... 77
11.3 Application to the SESAMO Use Cases ............................................................................................ 77
11.4 References ......................................................................................................................................... 81

Safety Analysis (IFAG and ESY) ................................................................................................................. 83
11.5 Motivation .......................................................................................................................................... 83
11.6 Description of Safety Criticality Analysis (SCA) ................................................................................ 83
11.6.1 SCA Prerequisites ............................................................................................................................. 84
11.6.2 Identification of Software structure ................................................................................................ 86
11.6.3 Safety Criticality Analysis .............................................................................................................. 87
11.6.4 HAZOP Analysis ............................................................................................................................ 88
11.7 SCA applied to Medical use case ....................................................................................................... 88
11.7.1 Description of the architecture ......................................................................................................... 89
11.7.2 Safety measures ............................................................................................................................... 91
11.7.3 Requirement decomposition and ASIL tailoring .......................................................................... 93
11.7.4 Safety Criticality analysis .............................................................................................................. 94
11.8 SCA applied to Automotive E-Motor use case ................................................................................ 97
11.8.1 Description of the architecture ......................................................................................................... 97
11.8.2 Safety level of the safety goals and allocation to software components ........................................ 97
11.8.3 Requirement decomposition for ASIL tailoring and safety measures ......................................... 98
11.8.4 Safety Criticality analysis .............................................................................................................. 99
11.9 Description of Dependant Failure Analysis (DFA) ........................................................................ 101
11.9.1 Types of dependent failure ............................................................................................................ 101
11.9.2 DFA Prerequisites ........................................................................................................................ 102
11.9.3 Identification of software elements or functions for DFA ........................................................... 103

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

11.9.4 Identification of Dependent Failures ................................................................. 103
11.9.5 Specification of Measures against Dependent Failures .............................. 105

11.10 DFA applied to Medical use case ................................................................. 105
11.10.1 Common Cause Failure Analysis ................................................................. 106
11.10.2 Cascading Failure Analysis ......................................................................... 106

11.11 DFA applied to Automotive E-Motor use case ........................................ 107
11.11.1 Common cause failure analysis ................................................................. 108
11.11.2 Cascading Failure Analysis ......................................................................... 109

11.12 Possible relation with security – Outlook .................................................... 110

11.13 Reference ........................................................................................................ 110
11.14 Vocabulary .................................................................................................... 111
11.15 Annex A. Communication fault model ....................................................... 111
11.16 Annex B. Mechanisms for error detection (ref. ISO 26262) ....................... 111

12 Conclusions ........................................................................................................ 112

Appendix

Preliminary Interdependency Analysis (PIA): Method and tool support ............. 109
Security-Informed Safety Cases ............................................................................ 165
Balancing security and performance ................................................................. 185
Safety versus Security in the Quality Calculus .................................................... 200
Automated Dependability Analysis of Complex Systems with the KB3 Workbench: the Experience of EDF R&D ................................................................. 220
Attack and Defense Modeling with BDMP .......................................................... 229
Modeling safety and security interdependencies with BDMP (Boolean logic Driven Markov Processes) ................................................................. 245
Towards Modelling Adaptive Attacker’s Behaviour ......................................... 255
LIST OF FIGURES

Figure 1: Overview of the PIA method and toolkit ................................................... ...................... 15
Figure 2: PIA method stages .................................................................................. 17
Figure 3: The KB3 workbench overall architecture ................................................... ...................... 36
Figure 4: First safety-security model for a medical device ........................................... ..................... 37
Figure 5: Algorithm for a deterministic attacker .......................................................... 45
Figure 6: a) the network system, b) the attack graph of the network system ................... 46
Figure 7: FMEA in relation to FTA ........................................................................... 50
Figure 8: Different FMEA Types[5] ........................................................................... 51
Figure 9: Risk Priority Number .............................................................................. 52
Figure 10: exemplary FMEA sheet ........................................................................ 55
Figure 11: Save Failure Fraction (SFF) as function of failure rates ............................... 57
Figure 12: SFF calculation sheet .............................................................................. 58
Figure 13: SFF calculation nomenclature .................................................................. 60
Figure 14: Categorization safety requirements ............................................................. 65
Figure 15: Generic use case of ESSI........................................................................ 78
Figure 16: Application specific use case of ESSI ......................................................... 78
Figure 17: Block schema of ESSI with relation to safety/security analysis ...................... 80
Figure 18: SCA within V-Cycle .............................................................................. 84
Figure 19: Structure of safety requirement (source: ISO 26262) ................................. 85
Figure 20: UML description of Medical use case ......................................................... 89
Figure 21: Allocation of safety level .......................................................................... 90

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Electricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

Figure 22: Tailoring of safety level ........................................................................................................94
Figure 23: E-Motor architecture ............................................................................................................97
Figure 24: Description of Common Cause Failure ..............................................................................101
Figure 25: Description of Cascading Failure ......................................................................................102
Figure 26: DFA within V-Cycle ..........................................................................................................103
Figure 27: Matrix for Common Cause Failure ....................................................................................100
Figure 28: Matrix for Cascading Failure ............................................................................................101

LIST OF TABLES

Table 1: Similarities between IMA and MILS requirements .................................................................32
Table 2: Results of satisfaction of security criteria ..................................................................................43
Table 3: C-FMEA, D-FMEA and P-FMEA[4] .......................................................................................50
Table 4: Severity of failure ranking .....................................................................................................53
Table 5: Probability of failure ranking ..................................................................................................53
Table 6: Likelihood of failure detection by design control .................................................................53
Table 7: SFF and resulting SIL .............................................................................................................56
Table 8: Medical - SCA functions grouping .........................................................................................95
Table 9: Medical - SCA failure modes .................................................................................................96
Table 10: E-Motor SCA Failure modes ...............................................................................................100
Table 11: Medical - Common cause failure ........................................................................................106
Table 12: Medical - Cascading Failure ................................................................................................107
Table 13: Automotive E-Motor - list of element for DFA .................................................................108
Table 14: Automotive E-Motor - Common Cause Failure ...............................................................109
Table 15: Automotive E-Motor - Cascading Failure ..........................................................................110
Table 16: Mechanisms for error detection at the software architectural level (ref: ISO 26262) .........111
EXECUTIVE SUMMARY

This document provides a “Specification of Safety and Security Analysis and Assessment Techniques: The preliminary version D3.1 will focus on the definition of the safety and security metrics, and on the definition of the methodologies and techniques for the assessment of the safety and security properties.” as detailed in the technical annex.

The document presents a number of core methodologies and techniques available among the partners, either in the form of well-developed tools or approaches that could be developed, based on the competences of the partners. It does not intend to present a comprehensive survey of all methods and techniques that exist in the field - rather it presents a selection of those that are likely to be most profitable to deal with the SESAMO use cases and to address the main challenge of SESAMO: to be able to deal with safety and security at the same time.

The core methodologies and techniques presented are:

- Stochastic models of interdependent infrastructures (CITY)
- Security-informed Safety Cases (CITY and ADEL)
- Message authentication and schedulability analysis on CAN bus (CTU)
- Pareto frontier between safety and security constraints (DTU)
- Safety and Security in the presence of Denial of Service Attacks (DTU)
- The KB3 Workbench (EDF)
- Safety & security analysis of resilient services in communication networks (FTW)
- Formal Metrics (IIT-CNR)
- State of the art FMEA techniques (SAG)
- Safety and security analysis for railway application (UC)
- Safety Analysis (IFAG and ESY)

The role of this document is to provide the necessary basis for the consortium to select, develop and apply those methods and techniques that seem most promising for dealing with the SESAMO use cases as identified in deliverable D1.1.

To facilitate this each section gives an overview of the method and techniques and then studies a number of use cases, making a preliminary feasibility study of which of the aims of the use case may be addressed using the proposed approach.

In the next period a more substantial testing of the ideas will be made and the up-coming deliverable D3.2 will reflect the choices of methods and techniques that seems most fruitful for reaching the objectives of SESAMO.

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, icky++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
1 INTRODUCTION

This document provides a specification of safety and security analysis and assessment techniques as detailed in the technical annex:

“Specification of Safety and Security Analysis and Assessment Techniques: The preliminary version D3.1 will focus on the definition of the safety and security metrics, and on the definition of the methodologies and techniques for the assessment of the safety and security properties.”

The document presents a number of core methodologies and techniques available among the partners, either in the form of well-developed tools or approaches that could be developed, bases on the competences of the partners. It does not intend to present a comprehensive survey of all methods and techniques that exist in the field - rather it presents a selection of those that are likely to be most profitable to deal with the SESAMO use cases. The final selection of approaches will be reflected in deliverable D3.2.

1.1 DOCUMENT STRUCTURE

After this introductory section we present several sections detailing core methodologies and techniques available within the consortium:

- The section "Stochastic models of interdependent infrastructures (CITY)" presents an overview of a method for modelling and analysis of complex systems including critical infrastructures. The method includes both – qualitative and quantitative aspect and progresses in steps from building qualitative models of complex systems of interacting components to constructing and solving via Monte Carlo simulation probabilistic models of complex systems of stochastically dependent components.

- The section "Security-informed Safety Cases (CITY and ADEL)" presents an overview on the impact that security considerations can have on a safety case. The main contribution is a methodology for extending safety cases to address security explicitly, for which we have started to develop tool support. The use of a structured safety case approach, based on Claims-Arguments-Evidence, makes it possible to explore trade-offs between safety and security requirements.

- The section "Message authentication and schedulability analysis on CAN bus (CTU)" presents an overview of the CAN bus and some of the protocols used there.

- The section "Pareto frontier between safety and security constraints (DTU)" presents an overview on how to balance considerations of security against performance using stochastic model checking. The main contribution is to evaluate different security policies against the risk of key compromise - focusing on resource bounded systems with limited cryptographic power - and identifies the design trade-off by means of Pareto frontiers. This provides an interface between the systems engineer and the quantitative security expert.
The section "Safety and Security in the presence of Denial of Service Attacks (DTU)" presents an overview on an approach to simultaneously model safety and security threats within the same programming notation. The main contribution is to support a programming style that is robust in the case of communication failure or denial of service attacks. This may provide a worthwhile path to explore in the absence of international standards dealing with the interplay between safety and security.

The section "The KB3 Workbench (EDF)" presents an overview of tools available within SESAMO for safety and security assessment. These tools are very general, domain independent and therefore could in principle be applied to most use-cases of SESAMO, provided the information on the systems to be studied is sufficiently detailed and precise.

The section "Safety & security analysis of resilient services in communication networks (FTW)" presents an approach to the analysis of communication networks.

The section "Formal Metrics (IIT-CNR)" presents a framework for a formal description of security metrics and an advanced analysis of attacker behaviour. The main contribution is the formal model of metrics, which allowed analysing whether the proposed metrics are able to measure security. Moreover, the advanced model of attacker behaviour (which uses some of the proposed security metrics) should provide a more accurate way to analyse strength of a security system.

The section "State of the art FMEA techniques (SAG)" presents Failure Mode and Effect Analysis (FMEA) as a main analysis technique in safety standards. The main contribution is to show in detail the benefits and methods to guarantee the required safety level. A practical FMEA example on the industrial drives use case will conclude the section.

The section "Safety and security analysis for railway application (UC)" presents an overview of safety and security analysis across whole RAMS life cycle of railway application and also provides overview of safety analysis methods applicable in railway use case with listing their advantages and constraints. This contribution identifies main functions of railway use case (provides secure channel and separation of open and closed transmission system) which will be analysed.

The section "Safety Analysis (IFAG and ESY)" presents a detailed description of the art of safety analysis methods used in the automotive domain in order to satisfy the functional safety standard ISO26262.

The deliverable concludes with a number of appendices that are typically longer presentations, perhaps research papers previously published or submitted for publication, that further expand on the presentation in one of the sections listed above.

1.2 TECHNICAL CONTEXT AND OBJECTIVES

The role of this document is to provide the necessary basis for the consortium to select, develop and apply those methods and techniques that seem most promising for dealing with the SESAMO use cases as identified in deliverable D1.1.

The selection of the methods and techniques reported upon has therefore been based on the competences of the partners, with preference for the more novel or well-developed directions available within the consortium, and dispensing with a treatment of methods and techniques that are standard.
within the area and not particularly addressing the primary challenge of SESAMO: the need to consider safety and security at the same time.

To facilitate the adoption of the methods and techniques presented, each section gives an overview of a particular approach. In some cases this is developed to some depth, and in other cases the more detailed development is referred to an appendix, in particular when they take the form of accepted or submitted research papers. It then goes on to study a selection of the SESAMO use cases and indicates some of the concrete aims that the methods and techniques might provide solutions to.

In this respect, the subsections detailing the applicability of the methods and techniques to the use cases, form the most important contribution of this report, because this is where a preliminary testing of the ideas is made. In the next period a more substantial testing of the ideas will be made and the up-coming deliverable D3.2 will reflect the choices of methods and techniques that seems most fruitful for reaching the objectives of SESAMO.
2 STOCHASTIC MODELS OF INTERDEPENDENT INFRASTRUCTURES (CITY)

Robin Bloomfield (CITY, Centre for Software Reliability and Adelard)
Oleksandr Nettachov (CITY, Centre for Software Reliability)
Peter Popov (CITY, Centre for Software Reliability)
Vladimir Stankovic (CITY, Centre for Software Reliability)

Summary. This section gives an overview of a method for modelling and analysis of complex systems including critical infrastructures. The method includes both qualitative and quantitative aspects and progresses in steps from building qualitative models of complex systems of interacting components to constructing and solving via Monte Carlo simulation probabilistic models of complex systems of stochastically dependent components.

The methodology presented in this contribution has initially been motivated by interdependency analysis of large complex critical infrastructures (LCCI) including national critical infrastructures. The methodology, however, is also applicable to other complex systems, e.g. distributed computing systems such as data centres or complex process control systems, which are not necessarily a part of national critical infrastructures.

2.1 MOTIVATION

One of the greatest challenges in enhancing the protection of Critical Infrastructures (CIs) against accidents, natural disasters, and intentional and unintentional disruptions is establishing and maintaining an understanding of the interdependencies between infrastructures and the dynamic nature of these interdependencies. Interdependency can be a source of “unforeseen” threat when failure in one infrastructure may cascade to other infrastructures, or it may be a source of resilience in times of crisis, e.g., by re-allocating resources from one infrastructure to another.

Understanding system complexity including interdependencies is a challenge both for governments and for infrastructure owners/operators. Both, to a different extent, have an interest in services and tools that can enhance their risk assessment and management to mitigate large failures that may propagate across infrastructures. However, cost of investment in infrastructure modelling and interdependency analysis tools and methods, including the supporting technology, may reach millions of pounds, depending on the size of the system to be modelled, on the level of detail and on the mode of modelling (real-time or off-line). These factors will determine the software, hardware, data and personnel requirements.

It is therefore very important to understand what the scope and the overall requirements of an analysis are going to be, before proceeding with such an investment. However, the decision on what modelling and visualisation capabilities are needed is far from simple. Detailed requirements may not be understood until some modelling and simulation has been conducted already, in order to identify critical dependencies and decide what level of fidelity is required to investigate them further.

Often techniques developed and successfully applied to dependability assessment are also used to study interdependency between CIs. Examples of such techniques are Fault Tree analysis, Petri nets, Markov chains, etc. These techniques may be difficult to scale up to the levels of complexity offered by CIs. Another practical difficulty with the analysis is that it typically may involve several
formalisms. A typical example would be combining the probabilistic analysis with a non-probabilistic analysis, e.g. Stochastic Activity Network with power flow models (i.e. deterministic models of either differential or merely linear equations) in power systems. In these circumstances conducting the analysis efficiently would require the chosen methodology to be supported by an adequate set of tools.

The Preliminary Interdependency Analysis (PIA) methodology attempts to address these challenges. It starts off at a high level of abstraction, supporting a cyclic, systematic thought process that can direct the analysis towards identifying lower-level dependencies between components of CIs. Dependencies can then be analysed with probabilistic models, which would allow one to conduct studies focussed on identifying different measures of interests, e.g. to establish the likelihood of cascade failure for a given set of assumptions, the weakest link in the modelled system, etc. If a high-fidelity analysis is required, PIA can assist in making an informed decision of what to model in more detail. The method is applicable as both i) a lightweight method accessible to Small-to-Medium Enterprises (SMEs) in support of their business continuity planning (e.g., to model information infrastructure dependencies, or dependencies on external services such as postal services, couriers, and subcontractors); and ii) a heavyweight method of studying with an increasing level of detail the complex regional and nationwide CIs combining probabilistic and deterministic models of CIs.

The PIA methodology is very generic in nature and could deal with stochastic processes generated by both accidental and malicious phenomena. In the past the methodology has been applied to cases in which only accidental failures/anomalies would be accounted for. Extending the studies towards malicious activities is a significant challenge. More specifically, the issues to be addressed are related to modelling the adversary (e.g. by building an adversary profile) and the effects of the attacks on the system properties of interest including security properties – availability of data, data integrity and confidentiality – but also how these affect other measures of interest, e.g. safety of the analysed systems.

2.2 OVERVIEW

Preliminary Interdependency Analysis (PIA) is an analysis activity that seeks to understand the range of possible interdependencies and provide a justified basis for further modelling and analysis. Given a collection of CIs, the objectives of PIA are to develop, through a continuous, cyclical process of refinement, an appropriate service model for the infrastructures, and to document assumptions about resources, environmental impact, threats and other factors.

PIA has several benefits. In particular, PIA can:

- help one to discover and better understand dependencies which may be considered as “obvious” and as such are often overlooked (e.g. telecommunications need power);
- support the need for agile and time-efficient analyses (cannot always wait for the high fidelity simulation);
- be also used by Small-to-Medium Enterprises (SMEs) and not just infrastructure owners and government.
PIA allows for the creation and refinement of models, in a focused manner, by revisiting earlier stages in the PIA process in the light of the outcomes of latter stages. For example, an initial application of PIA should result in a sufficiently concrete and clearly defined model of the system being analysed and the parts thereof (including their dependencies). However, following the first design iteration, an analysis of the model could cause us to question the assumptions made earlier on in the design process. As a consequence, the model may be revised and refined; as we shall see later on, revisiting previous phases of the design process is a key aspect of the PIA method and philosophy overall.

PIA consists of two parts:

- **Qualitative analysis.** The modelling exercise begins with a definition of the boundaries of the system to be studied and its components. Starting off at a high level, the analyst may go through a cyclical process of definitions, but may also be focused on a particular service, so the level of detail may vary between the different parts of the overall model. The identification of dependencies (service-based or geographical) will start at this point.

- **Quantitative analysis.** The models created during the qualitative PIA are now used to construct an executable, i.e. a simulator of the model behaviour in the presence of failures of the modelled entities for the chosen model parameterisation. The model parameterisation may be based either on expert judgment or on analysis of incident data.

The PIA Toolkit provides support for both the qualitative and quantitative analyses. Figure 1 illustrates an overview of the method and the toolkit.
The interdependency models, of course, have to be related to a purpose and this should be captured in terms of a scenario and related requirements. The narrative aspect of the scenario is enormously important as it provides the basis for asking questions and discovering interdependencies as the starting point for more formal models.

Typically the systems of interdependent CIs of interest are complex: they include many services which in turn consist of many parts. Given the complexity and size of the analysed systems tool support is essential.

### 2.2.1 Safety and Security metrics

There is a range of metrics used to measure the resilience of CIs, including the impact of interdependency. These are typically various *loss functions*, e.g. the “size of cascade” (i.e. the number of components that are simultaneously down), the duration of a cascade or a combined measure of loss (e.g. the product of the cascade size and its duration). These metrics do not discriminate between the causes of the losses. The dominant view among the operators of CIs is that currently the accidental failures are the main concern. On the other hand the importance of losses caused by malicious behaviour – be it vandalism (or other forms of breaching physical security) or cyber attacks
has not recognised and it is acknowledged that the importance of these losses is increasing, e.g. with regard to “smart grids/cities” where telecommunications play a significantly more important role than in the traditional infrastructures.

Metrics commonly used to characterise security are related to availability, integrity and confidentiality of information: we could quantify these using probabilistic measures such as the probability of data being available, the probability that data is accurate (i.e. is not tampered with or erased) and the probability of data not being stolen/accessed by a non-authorised party.

The challenge here is two-fold:

- How one can get accurate estimates of these probabilistic measures? The main difficulty is that the behaviour of the adversary may vary greatly over time, thus even accurate measurement in a particular environment may be a poor predictor of what to expect in other environments;
- How can one quantify the impact of various cyber attacks on the losses that are of interest in the specific system context? For this aspect we need models of the impact of the individual successful attacks on a CI which are context specific (e.g. may be affected by the mode of operation of the CI at the time of a successful attack, etc.). Although conceptually this problem is not particularly new – similar concerns apply to accidental failures too – the sheer scope for possible attacks makes it very difficult to apply quantitative risk assessment here. Taking a conservative view, e.g. assume the worst possible scenario for every attack, is not particularly useful as one will conclude soon that there is very little that can be done.

Our approach to extending PIA methodology to dealing with cyber attacks will be pragmatic – we will start with the specific scenarios defined by the case study providers and in the process of building the system models we will explore other plausible scenarios worth investigating.

2.2.2 Overview of the technical paper “Preliminary Interdependency Analysis (PIA): Method and tool support”

This source is a technical report produced as deliverable in the UK Technology Strategy Board funded project PIA:FARA. The report presents a method called Preliminary Interdependency Analysis (PIA) which can be used to conduct interdependency analyses in complex systems such as Critical Infrastructures. The method is supported by a toolkit which can be used to conduct qualitative and quantitative analyses of interdependencies between complex systems.

The report contains two parts. The first one describes the PIA methodology – how one builds a qualitative model of the system to be studied. Using a HAZOP-style of analysis one identifies the important (inter)dependencies, which itself is usually very useful. In case quantification is needed, the method requires that a large number of probabilistic parameters be obtained (e.g. using the available data sheets or via direct measurements) which will allow for solutions to be derived using Monte-Carlo simulation. The second part describes the tool chain that supports the methodology. The tool support offers an easy way of extending the pure probabilistic models with deterministic ones (e.g. the power flow models mentioned above) and thus allows for studies with different degrees of fidelity using Monte Carlo simulation.
Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
2.3 APPLICATION TO THE SESAMO USE CASES

2.3.1 AKHELA

The Akhela use case contains three scenarios that focus on analysing and mitigating the risks of cyber-security attacks and handling the repercussions of the incidents. The scenario offers scopes for interdependency analysis of two infrastructures - the oil refinery and the respective power system.

The case study offers a system that consists of two sub-systems - the refinery and the power sub-system. The model of the refinery, in turn, consists of ICT components (computers running specialised software) and hundreds of tightly coupled interconnected devices (actuators and sensors) via which the refinery processes are controlled. A model of the refinery will be built at a suitable level of abstraction accounting for the important dependencies between the components. The model will account for both the accidental failures of the components and malicious cyber-attacks and how these may affect the overall operation of the refinery (suitable loss functions will be defined).

The PIA methodology supports the analysis by providing the tools needed for modelling the elements of the system, establishing *stochastic associations* between them (e.g. via functional or spatial dependencies) and performing a quantitative analysis of the safety and security in separation or of the system resilience. The findings obtained with the models can be used: i) proactively, e.g. to identify the “weakest link” in the system and improve the system’s overall resilience by targeted investment (e.g. using redundancy/diversity to improve the “weakest links”); ii) reactively, e.g. to analyse/stress the various plans that might exist for business continuity or emergency response in case of incidents.

This case study offers a complex scenario in which both the accidental and malicious failures are present and thus will serve as a suitable testbed to test the planned extensions of the PIA toolkit to deal explicitly with cyber attacks.

2.4 POSSIBLE APPLICATIONS TO OTHER SESAMO USE CASES

2.4.1 EDF

This use case as it stands is not ready for any quantitative analysis. The proposed 3 examples of malicious activities compromising safety can be modelled using PIA, but the value of such an effort is as yet unclear.

There has been interaction with colleagues from EDF with a view to identifying a suitable case study. If this effort is fruitful, we could apply the PIA tool to the EDF case study, too.

2.5 APPENDIX

3 SECURITY-INFORMED SAFETY CASES (CITY AND ADEL)

Kateryna Netkachova (Centre for Software Reliability, City University London)
Robin Bloomfield (Adelard LLP and City University London)
Robert Stroud (Adelard LLP)

Summary. This section gives an overview on the impact that security considerations can have on a safety case. The main contribution is a methodology for extending safety cases to address security explicitly, for which we have started to develop tool support. The use of a structured safety case approach, based on Claims-Arguments-Evidence, makes it possible to explore trade-offs between safety and security requirements.

3.1 MOTIVATION

A detailed and comprehensive safety case is a must for any safety-related system. Safety cases are explicitly required by regulations and standards across a wide range of industries. A safety case should consist of a structured argument, supported by a body of evidence that provides a demonstrable and valid argument that a system is adequately safe for a given application in a given operating environment.

In order to analyse complex safety- and security-critical embedded systems, an approach to joint analysis of safety and security properties needs to be developed. We believe the best solution is to enhance the existing safety case methodology and use it to demonstrate and communicate security requirements in addition to safety. In this way we can benefit from a mature, effective and time-tested safety case approach and have both safety and security properties considered in an integrated manner within a security-informed safety case. This approach should also give us a better understanding of the interactions between system safety and security aspects and help us to address the potential issues associated with their interrelation. The development of such an approach requires research into methodology, tools, and use cases.

3.2 SCIENTIFIC CONTRIBUTION

Introduction and general concepts

In the broad context of a generic understanding of safety, it would be appropriate to consider system security to be part of the overall safety case if we are dealing with both security- and safety-critical systems. However, it is worth bringing security aspects explicitly into focus by constructing a security-informed safety case for a system. This will highlight the role of security in the system safety justification and ensure that both security and safety properties are addressed in the resulting case. A security-informed safety case will be more representative and fit for purpose in analysing security- and safety-critical systems.

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
The main goal of our current work is to develop a methodology for creating security-informed safety cases with tool support. We want to achieve a usable approach that can deal with the safety and security properties of a system in an integrated way and ensure that the system is safe and operable in a given environment, against a specific set of security threats.

Our solution is to develop an enhanced structured safety case methodology that jointly address safety and security aspects and their interrelationships for embedded systems in multiple domains.

We also propose to develop tool support for structuring and assessing arguments, and feeding evidence into the claim structure, in order to assist in the creation and management of security-informed safety cases.

Basic safety case structure

Our approach is based on the Claims-Arguments-Evidence (CAE) formalism, which is a simple yet effective notation for structuring arguments to communicate how a system is adequately safe in its environment. The basic idea is that we make a claim, and this claim must be supported by evidence through a valid and structured argument. The first question is what sort of claim we want to make. For example, we might make a top-level claim that the system is acceptably safe and operable, and then break it down into a set of claims about different attributes of the systems.

In general, we want a system to satisfy certain principles: it needs to be a well-designed system, it needs to comply with standards, and of course, it should have the required behaviour.

Therefore, the justification should address:

- The behaviour of the system;
- The design principles that were used to implement it;
- Compliance with standards and regulations.

Each of these claims can be further decomposed into a set of sub-claims. For example, in terms of behaviour, we are basically interested in ensuring that the system only does good things and does not do bad things. So we need to make additional claims to analyse if our system satisfies its safety properties (positive properties) and whether we have mitigated all vulnerabilities and hazards (negative properties).

Introducing security into safety analysis

The CAE approach to developing and structuring safety cases has been enhanced in recent years and part of the methodology is the triangle of assessment: positive properties (safety behaviours), negative properties (hazards and vulnerabilities), and compliance.

When analysing safety- and security-critical systems, additional security related aspects should be introduced to the triangle. In particular, the triangle should be adjusted as follows: positive properties (safety and security behaviour), negative properties (hazards, threats and vulnerabilities), and compliance with both safety and security standards and regulations. This approach is illustrated in Figure 1 below.
Using this approach, safety and security assurance can be explained in terms of the updated triangle as follows:

- Justification via a set of claims/goals about the system’s security and safety behaviour;
- The use of accepted security and safety standards, best practices and guidelines;
- An identification and investigation of safety hazards, security threats and vulnerabilities of the system.

**Impact of security on claims and arguments**

Security considerations are needed to enhance the analysis and ensure that the system is operable, safe and also secure for a given application in a given environment, against a specific set of security threats. We start with a simple safety case structure and proceed to analysing what kind of impact security has on the safety case. The idea is to discover how the existing claims and arguments are affected by security considerations, and whether any new claims and arguments are necessary. To analyse how the case is affected by security we need to go through the claims, arguments and evidence, review each of the elements and decide what needs to be done in a security context.

We start by viewing the safety case with respect to the security attributes: confidentiality, integrity, availability. Integrity and availability are dealt with when analysing safety, so they are usually already within the case. Confidentiality is almost never a part of a safety case so it has to be added and considered in detail. There are two kinds of confidentiality properties that need to be addressed. First, we might want to protect system assets from disclosure as a matter of principle. Second, there is a concern that disclosing information about a system might enable the system itself or related systems to be attacked. For example, if hackers know the design or particular techniques that were used in the system, they might be able to design attacks in a better way. So there is a new confidentiality related claim that needs to be added to say that the system does not leak any information that can be used to enable attacks on other systems.
The other security-related claims that need to be considered are concerned with deployment, design, interactions (e.g. related to keys or different types of protocols), configuration, training, and vulnerabilities.

Figure 2 demonstrates an example of the impact of security on the top part of a basic safety case. The red colour highlights nodes that were affected by security.

Figure 2. Outline impact of security on a top part of safety case

It is also important to consider the impact of security on claims and arguments about the future behaviour of the system. Here, the goal is to ensure that the system will continue to be operable, safe and also secure throughout its lifetime.

There are a number of possible events that the system may need to deal with in the future. Such events include failures of system components, changes to the environment, discovery of new vulnerabilities, and so on. The handling of these events requires adjustments to address security in addition to the safety related issues that are normally considered. We need to demonstrate that security is properly analysed, all possible future events that may pose security risks are considered and that the system responds appropriately.

The key objective is to ensure that the system will continue to behave as required, preserving both safety and security properties in the face of various events. Therefore a new claim such as “Security properties are maintained throughout the lifetime” needs to be introduced.
In order to support this claim, both proactive and reactive security measures should be implemented in the system. Proactive measures are employed to protect the system from intruders and prevent malicious attacks, while reactive measures are needed to respond to security issues and remedy the situation when the system is already compromised and attacks occur.

Consideration of all these factors leads to extended argumentation and additional sub-claims.

Additionally, periodic security audits (both internal and external) are important to analyse the adequacy of security measures.

And finally, it is necessary to have appropriate procedures to handle security incidents in order to keep the system operable or recover it quickly in the event of attack. Therefore a claim like “Incident handling is performed” should be added to the case and expanded further by considering whether the emergency response to security attacks is appropriate, all detected security issues are made available to the developers and the proper procedures to resolve discovered issues are defined and performed.

A more detailed description of the impact of security on the safety claims is provided in an appendix.

Tool support

We plan to develop tool support for our methodology. In particular, we wish to explore how to:

- build security-informed safety cases more efficiently
- link our cases to models with the potential for more rigorous reasoning

We have started developing a software tool for harvesting evidence and dealing with compliance. This is an evaluation tool based on a security and safety questionnaire, which should help us to analyze whether a system conforms to various known safety and security standards. Additionally, a plugin is being developed for the ASCE assurance case tool that will enable us to import the evidence generated by this evaluation tool into a security-informed safety case.

3.3 APPLICATION TO THE SESAMO USE CASES

We analysed the impact of security on a hypothetical safety case, recording observations and findings that we deemed to be of use.

We now plan to interact with various use case providers to communicate and refine our approach in order to make it as useful and efficient as possible.

An important aspect we have learnt from our interaction to date is that many partners rely on a compliance case rather than a behaviour-based case. We will need to explain how behaviour-based cases and compliance cases relate to each other and to do some detailed work.

The main use cases we plan to apply our approach to are listed below:

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

- Integrated Modular Avionics (IMA) use case from EADS
- Oil and Gas use case from Akhela
- Medical Applications use case from Infineon and eesy-id

To date, only the collaboration with EADS and SYSGO has been worked out in detail. The other use case providers have expressed their interest; however, the details of the collaboration are yet to be decided.

Avionics. We plan to do a case study with EADS and SYSGO on secure gateway. We propose to develop a security-informed safety case for the gateway architecture and see two main benefits to this collaboration:

1. Structured assurance cases are useful for communicating and building confidence in the safety and security properties that are being claimed. By developing a security-informed safety case for the secure gateway, we will support EADS in exploring the trade-offs and interactions between safety and security, and identify any conflicts between safety and security standards.
2. By developing an assurance case for the secure gateway, we will provide EADS with an independent review of the architecture. As part of our analysis, we will develop attack scenarios and abuse cases that explore potential security vulnerabilities and ensure that appropriate security controls are in place. In this way, we will help EADS to clarify the architecture requirements for the secure gateway.

The main directions we plan to focus our effort on are:

- Applying the security-informed safety case methodology to a specific example, namely, the design of secure gateway architecture.
- Investigating the extent to which following a methodology such as DO178B/C, and more specifically, ED202 / ED203 addresses security concerns as well as safety concerns.
4 MESSAGE AUTHENTICATION AND SCHEDULABILITY ANALYSIS ON CAN BUS (CTU)

Michal Sojka (Czech Technical University in Prague)

Summary. This section gives an overview of previous work in the area of message authentication on the CAN bus and identifies missing features that are needed for proper use of message authentication on CAN bus in SESAMO eMotor use-case.

4.1 MOTIVATION

CAN bus is a commonly used communication in cars. Although it is being slowly phased out by more modern networks such as FlexRay or Ethernet, it will definitely remain in cars for several years from now. CAN is a broadcast bus and does not provide any authentication features. Any device on the bus can therefore pretend to be another device and receivers cannot distinguish who sent the particular message.

As today’s cars are more and more connected to the outside world, it is more likely that certain ECUs can be remotely compromised [3]. Without the authentication on the CAN bus the compromised ECU can easily pretend to be another ECU, which can lead to the compromise of the whole car.

For this reason, several solutions were developed that implement message authentication on the CAN bus. There, a message authentication code (MAC) is transmitted together with the data of the message. MAC can be used by the receiver to authenticate the sender. Usually, MAC is the result of some cryptographic computation based on either a shared secret or public key infrastructure. The main problem with appending MACs to the CAN messages is the limited maximum payload of a CAN frame (8 bytes) and limited CAN bus bandwidth (1 Mbit/s). In automotive, busses are usually clocked even lower: either at 500 or 125 kbit/s.

Herrewege et al. developed CANAuth [4] – a protocol that is based on hardware-extended CAN bus called CAN+. This extension allows transferring more data within a single frame so that the MAC can be easily added to every frame. This solution is partially compatible with existing hardware but only CAN+ compatible hardware will be able to read the MAC. This limits the adoption of this solution by the automotive industry. Similar to CAN+, the new standard “CAN with Flexible Data-Rate” (CAN FD), developed by Bosch, can also transfer bigger payload in a single frame and this can be used for adding MACs.

Several other solutions use an existing CAN hardware and add authentication at the application protocol level. Schweppe et al. [2] developed a solution where a single message is fragmented into several CAN frames in order to include the MAC. This leads to bigger delays in communication, which might not be acceptable for real-time or safety relevant communication. Later Hartkopp et al. [1] developed a solution that allows MACs to be transmitted in the same CAN frame as the data. The

Project Partners:

Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
limitation of their approach is that the maximum size of the authenticated signal is 32 bits and the MAC is only 32 bits in size. This means that the chance of guessing the secret keys used to generate the MAC is higher than for other approaches that allow longer MACs to be used.

4.2 **SCIENTIFIC CONTRIBUTION**

In our work, we concentrate on protocols described in [1] and [2] because these approaches can be used directly with existing hardware. As can be seen there is a clear conflict between safety (meeting of communication deadlines) and security (the size of MAC). Increasing the size of MAC improves security (lowers the chance of attacker guessing the key used to generate the MAC) but requires more bandwidth and hence may lead to violation of safety properties. In most cases designers want to maintain the safety properties and add as much of security as the bus bandwidth allows. The level of security is represented by several parameters such as the size of the used MAC, the frequency of authenticated messages, the resolution of clocks to avoid reply attack etc. In the previous work, these parameters were determined via ad hoc methods based on engineering intuition and simple simulations. We work on an analysis method that allows designers to calculate the optimal values of security parameters based on the requirements and provides guarantees that safety properties are met.

4.3 **APPLICATION TO THE SESAMO USE CASES.**

Infineon’s eMotor use case requires integrity verification of requests coming from the network. Besides providing integrity, MAC also provides the authentication of the sender and can thus prevent certain attacks on eMotor infrastructure.

**References**


5 Pareto Frontier Between Safety and Security Constraints (DTU)

**Summary.** This section gives an overview on how to balance considerations of security against performance using stochastic model checking. The main contribution is to evaluate different security policies against the risk of key compromise - focusing on resource bounded systems with limited cryptographic power - and identifies the design trade-off by means of Pareto frontiers. This provides an interface between the systems engineer and the quantitative security expert.

5.1 Motivation

Security is an overloaded word in the information technology realm, and it may have different meanings depending on context. No matter which context is used, there is one major fact: *security is not for free!* Security comes with certain costs, which could be associated to performance, energy, computation, memory, or even life-time of a device. If these costs for desired security level exceed the budget, then either the resources should be enhanced or, as often it is the case, a degree of security should be sacrificed.

Security experts are often struggling to find configurations that provide a suitable balance between the level of security and the performance. In today's world of smart and mobile devices, performance is highly related to the energy consumption.

Designers and engineers constantly try to design and implement more and more energy-efficient systems. In the lower levels, the aim is to design energy-efficient integrated circuits as well as benefit from energy harvesting. In the higher levels, applications are trying to make best use of the energy budget.

For such resource-constrained networks, updating security keys is one of these bottlenecks: if it is performed too often, then the energy requirement might be too high; on the other hand, if it is performed too seldom, then the risk of the network being compromised might be too high.

5.2 Scientific Contribution

We outline an approach to use stochastic model checking to obtain information about the interplay between levels of security and actual performance. Our running example considers different strategies for updating security keys in a network of sensor nodes: if it is performed too often the energy requirement might be too high; on the other hand if it is performed too seldom the risk of the net-
Dissemination level: PU

work being compromised might be too high. We develop a formal model for such scenarios and investigate the Pareto-frontier between security and performance, thereby providing application designers with information about the possible trade-offs. We also consider how to align this approach with the consideration of safety, which in the context of our running example involves that the devices continue operating without the presence of (possibly internal) failures. We conclude by discussing the interplay between stochastic model checking with multiple reward structures and the solution of fuzzy constraints to describe degrees of compliance with safety and security measures.

Overview of “Balancing Security and Performance” included in the Appendix. The above introduction is largely taken from the paper Balancing Security and Performance that is included in full in the Appendix. Section 2 defines the key update problem in a resource-constrained setting, Section 3 presents several key update strategies that are suitable for such settings, Section 4 provides a quantitative approach that introduces probabilities and metrics to the key update problem, Section 5 is an appetizer to our formalization and quantitative verification using stochastic model checking, Section 6 presents the design-efficiency curves at work, Section 7 discusses related technical issues, and finally Section 7 concludes.

5.3 APPLICATION TO THE SESAMO USE CASES

We then consider a few use cases from D1.1 and analyze the extent to which these ideas address problems of relevance to the use cases.

E-motor. In this use case, we spot certain resource constraints such as timing, bandwidth, etc. These constraints make it more challenging to achieve security guarantees. Our work on Balancing Security and Performance is relevant in this context, because our way of dealing with design space and achieving an optimal security solution within the resource-budget can be applied to the e-motor use case. For example, we may provide the designers with valuable information such as design-efficiency curves focusing on the trade-off between security and bandwidth, as well as security and timing, where security will be relating to one or more certain properties such as confidentiality, integrity, etc.

Medical. In this use case, we observe certain safety and security requirements that would require building blocks such as encryption, access control, integrity protection etc. However, these protection mechanisms will come with certain costs, and they might need to be balanced in many cases. For example, the patient monitors that are used to observe vital signals such as heart rate, oxygen concentration, and blood pressure are using wireless communication to enable the patient a valuable level of mobility, such as walking/moving with the portable part of the patient monitor. Such devices have limited battery lifes, e.g. 2 hours, and before the battery completely drains they need to be recharged on their dock (which could be the body of the patient monitor). The transmission from this monitor is important, because it sends signals to the emergency nurse or doctor on the life signals of the patient. Adding security and privacy to such a wireless patient monitor is of course meaningful, but on the other hand it will shorten the battery life. Therefore, there needs to be a balance between the level of security (and privacy) and the energy consumption (and battery life) of the device. From this perspective, our work on Balancing Security and Performance provides a framework that can be applied to this medical use case. Besides, there is an obvious trade-off between safety and security, because increasing the level of security in such a patient monitor could jeopardize the safety of the patient.
Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
6 SAFETY AND SECURITY IN THE PRESENCE OF DENIAL OF SERVICE ATTACKS (DTU)

Hanne Riis Nielsen (DTU Applied Mathematics and Computer Science)
Flemming Nielson (DTU Applied Mathematics and Computer Science)
Alessandro Bruni (DTU Applied Mathematics and Computer Science)
Ximeng Li (DTU Applied Mathematics and Computer Science)
Ender Yuksel (DTU Applied Mathematics and Computer Science)

Summary. This section gives an overview on an approach to simultaneously model safety and security threats within the same programming notation. The main contribution is to support a programming style that is robust in the case of communication failure or denial of service attacks. This may provide a worthwhile path to explore in the absence of international standards dealing with the interplay between safety and security.

6.1 MOTIVATION

One of the challenges of cyber-physical systems is to reconcile the often conflicting demands of security and safety. Safety concerns making sure that no harm can arise from using the systems; as an example an airplane should continue flying, and a car should continue to react to braking and steering. Security concerns making sure that nobody can pretend to be somebody else; as an example, steering directions should originate from the steering wheel and not from a car game played by the children on the back seat. These demands often conflict because safety requires very fast reaction to alerts whereas security requires that sufficient time is taken to ensure the authenticity (or integrity) of the alert. Yet both demands need to be addressed in order for the overall cyber-physical system to live up to expectations and hence intelligent trade-offs are called for.

The design of cyber-physical systems grows out of the design of embedded systems and is often performed by companies having a strong safety culture. They are often well trained in using general safety standards as well as more specialised safety standards for automotive, aeronautic, or health care applications. As communications increasingly become wireless or are multiplexed over optical fibers or electrical wires, there is a clearly identified need to incorporate security in order to prevent cyber attacks on systems. While it is recognized that both safety and security are important, one often hears the slogan that in the time of crisis safety takes precedence over security.

We believe that this slogan reflects an unclear understanding of the many-faceted nature of security. We might agree that in the time of crisis safety takes precedence over confidentiality; it would be imprudent not to communicate warnings about safety problems merely because these warnings can be overheard by others. However, to challenge the slogan of the safety community, we would like to suggest that authenticity (and integrity) takes precedence over safety; it would be imprudent not to check that orders to change course do in fact originate from authorised sources rather than (unaware or malicious) intruders.

Another maxim of the safety culture is that systems should be fail-safe; this means that systems either do not fail or that they fail in a particularly graceful manner having as few safety implications as possible. Taking the unreliability of communication into account this means that entire systems are designed around instructions of the form “please feel free to continue reversing (at the current or
lower speed) for the next 3 seconds” rather than “please feel free to continue reversing (at the current or lower speed) until instructed otherwise”, because if communication breaks down the former would lead to much fewer safety incidents than the latter; the design of the European Railway Train Management System (ERTMS) is a case in point.

This directly relates to the risk of denial of service as considered in security. While attacks on the confidentiality, integrity and authenticity of messages can be averted through the proper use of cryptographic communication protocols, there is hardly any feasible way to guarantee against denial of service attacks in cyber-physical systems. The reason is that cyber-physical systems are by their very nature open and hence wireless communication can be jammed and optical fibers and electrical cables can be cut. This suggests that the proper way to deal with denial of service attacks is to ensure that systems are developed in such a way that the consequences of communication breakdown are as benign as possible.

6.2 SCIENTIFIC CONTRIBUTION

In a previous paper we proposed a process calculus, the Quality Calculus, that allows to express due care in always having default or substitute data available in case the real data cannot be obtained due to unreliable communication. The development was facilitated by a SAT-based robustness analysis to determine whether or not undesirable error configurations could in fact be avoided by always choosing alternative configurations possibly using default or substitute data. This addresses the issue of how best to secure systems against denial of service and to obtain overall fail-safe behaviour.

In this paper we extend the Quality Calculus with trust levels indicating the degree of trust we can have in the authenticity and integrity of communications. In the interest of simplicity (and in accordance with our slogan that authenticity takes precedence over safety) we shall ignore all issues relating to confidentiality of communications. Instead we shall develop analyses to identify the extent to which a system is robust against the following two types of attacks:

- denial of service attacks due to other processes experiencing faults of their own or due to an attacker disrupting the communication, for example by jamming, and
- integrity attacks due to an attacker breaking part of the cryptographic communication protocol, for example by a brute force attack on a weak crypto-system or through the recovery of current session keys by physically dismantling and inspecting devices in the cyber-physical system.

Our aim is to identify those points in systems where decisions are made to increase the integrity level, either through

i. the endorsement of data (so as to allow an explicit flow of information otherwise prohibited), or through

ii. the temporary assertion of a higher trust-level for the context (the “program counter” so as to allow an implicit flow of information otherwise prohibited),
and to identify those points where the proper use of default or substitute data succeeds in

iii. establishing the non-reachability of certain “error” configurations.

We do so by introducing language primitives to indicate the three points of consideration and a safety and security type system enforcing that points of type (i) and (ii) have not been forgotten and that points of type (iii) have not be introduced without due care. The safety and security system amalgamates an adaptation of the SAT-based robustness analysis of with type systems for integrity, which are dual to type systems for confidentiality.

6.2.1 Overview “Safety versus Security in the Quality Calculus”

The above introduction is largely taken from the paper Safety versus Security in the Quality Calculus that is included in full in the Appendix. Section 2 defines the syntax of the Quality Calculus, Section 3 defines the semantics in both closed and open environments, Section 4 defines the safety and security type system, Section 5 contains a worked example, Section 6 discusses the theoretical properties of the system, and finally Section 7 concludes.

6.3 APPLICATION TO THE SESAMO USE CASES

We now consider a few use cases from D1.1 and analyze the extent to which these ideas address problems of relevance to the use cases.

Avionics. The Avionics use case combines safety considerations as expressed by IMA (Integrated Modular Avionics) and security considerations as expressed by MILS (Multiple Independent Levels of Security). It is interesting to observe that there is a rather close correspondence between some of the key requirements of IMA and MILS as summarised in the following table.

<table>
<thead>
<tr>
<th>IMA Requirements</th>
<th>MILS Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time and space partitioning</td>
<td>Separation</td>
</tr>
<tr>
<td>Intra/inter-partition communication</td>
<td>Controlled information flow</td>
</tr>
<tr>
<td>Mixed safety criticality levels</td>
<td>Multiple levels of security (trusted code, untrusted code, etc.)</td>
</tr>
<tr>
<td>The duty of the partitioning kernel and of the programmer</td>
<td>The duty of the separation kernel and of the application designer</td>
</tr>
<tr>
<td>Replication and spare modules</td>
<td>Non-by-passable and tamper-proof</td>
</tr>
</tbody>
</table>

Table 1: Similarities between IMA and MILS requirements

The work on Safety versus Security in the Quality Calculus provides a uniform frame for the co-consideration of security and safety but only in the qualitative domain (meaning that no time or probabilities are taken into account). The security levels of MILS are directly captured by the trust levels of the trust lattice as used in the Quality Calculus. The safety considerations of IMA are only dealt with to the point of expressing the non-reachability of undesirable code.
The strong point of the development is that it is accompanied with automatic analysis techniques for determining the trust annotations on control actions performed and for verifying that undesirable code is indeed unreachable. Technically this is performed using state-of-the-art SAT solvers for efficiently determining the satisfiability of Boolean propositional formulae.

For a full treatment we would need an extension with considerations of time, in particular real time, but perhaps also stochastic time. This is a challenge for state-of-the-art but one that we plan to start addressing in relationship to the SESAMO project as well as the MT-LAB research centre (a VKR Centre of Excellence studying the Modelling of Information Systems).

Other possible extensions include the incorporation of additional safety considerations and perhaps transferring the approach from modeling notations (such as the Quality Calculus) to programming notations (such as C).

**E-motor.** In the e-motor use case, security attacks could potentially influence the car control network from the infotainment network since both are connected. Hence there will be trusted inputs originating from within the car control network and untrusted inputs from the infotainment network, to the Complex Device Driver software components. Further, in case the car tuner has direct access to the car control network, there could also be different trust levels of data sources within the car control network, depending on the relative ease of tuning different parts. In the second case the control logic will then receive inputs at different trust levels. Moreover, at the same time we should consider the possibility of unavailable inputs to the Complex Device Driver software components due to failures of data sources, thereby guaranteeing fail-safe behaviors.

These considerations re captured by the binders of the Quality Calculus that allow consideration for partial unavailability of inputs. This helps to ensure that undesirable situations are indeed taken into consideration, and situations where it is impossible to provide safe behaviors might fortunately be unreachable in the code.

The strength of the development is that the Quality Calculus is accompanied with automatic analysis techniques (SAT solving) for determining the trust annotations on control actions performed and for verifying that undesirable code is indeed unreachable.

Extensions would be needed in order to also study the timing issues involved in this use case with the Quality Calculus.

**Car Infotainment.** The Car Infotainment use case contains a nice mix of confidentiality, integrity, availability and safety challenges. Confidential data provided to mobile devices must not leak; applications downloaded by the mobile device must be trusted; information exchanged between the car and the mobile device must not be modified by an untrusted third party; malformed data must be controlled; connectivity to mobile devices must be available as often as possible; remote control should be limited to intended features and only authenticated users should be able to control specific functionalities (e.g. the driver’s seat position should not be adjustable by anybody else).

---

**Project Partners:**
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ik++ Technologies, eeye-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
The development of *Safety versus Security in the Quality Calculus* directly captures the interplay between integrity and safety considerations. The trust levels from the trust lattice would be immediately suitable for expressing the integrity considerations needed in the use case. While there is no consideration of confidentiality there is a well known duality between integrity and confidentiality that would make it straightforward to add confidentiality annotations as well – perhaps inspired by the approach of the Decentralised Label Model. Our worked example of using the CD drive for multiple purposes (infotainment and system update) is directly applicable to the concerns of the use case. It would be easy to generalise to other mechanisms for system update than a given car manufacturer would prefer to use instead of a CD drive.

The strong point of the development is that it is accompanied with automatic analysis techniques for determining the trust annotations on control actions performed and for verifying that undesirable code is indeed unreachable. Technically this is performed using state-of-the-art SAT solvers for efficiently determining the satisfiability of Boolean propositional formulae.

**Medical.** The Medical use case considers Mobile Ambient Assisted Living Systems (AAL) intended to supervise patients remotely; the system is designed as a multi-purpose data concentrator which gives rise to considerations of safe and secure data concentration and safe and secure communication. It is important to guard against data being revealed to an unauthenticated third-person, one should be aware that information leaked may be used to derive methods to change the system configuration, there is a risk of denial of service on the data transmission, abnormal input may affect security relevant software elements, safety mechanisms may fail to detect unsafe system state, and there may be insufficient redundancy in the system.

The development of *Safety versus Security in the Quality Calculus* addresses the authentication considerations through the consideration of trust levels from the trust lattice. Given the duality between integrity and confidentiality it would be relatively straightforward to modify the approach to deal with confidentiality and privacy. We have carried out studies of different redundancy protocols in the context of the Quality Calculus in order to achieve a desired level of safety; here redundancy is used as the main defence against hardware failures.

The strong point of the development is that it is accompanied with automatic analysis techniques for determining the trust annotations on control actions performed and for verifying that undesirable code is indeed unreachable. Technically this is performed using state-of-the-art SAT solvers for efficiently determining the satisfiability of Boolean propositional formulae.

**Railway Communication.** The Railway Communication use case mainly focuses on security in a rather narrow context of on-board electronics. The development of *Safety versus Security in the Quality Calculus* does not directly address these concerns but would be relevant if the Railway Communication use case was considered to include the context in which the system operates.
7 THE KB3 WORKBENCH

Marc Bouissou, (EDF R&D, MRI department)

Summary.

EDF proposes its already available tools (the KB3 workbench) for safety and security assessment. These tools are very general, domain independent and therefore could in principle be applied to most use-cases of SESAMO, provided the information on the systems to be studied is sufficiently detailed and precise.

7.1 MOTIVATION

The modeling approach associated to KB3 relies on knowledge bases written in the Figaro modeling language. Figaro is domain independent, but the knowledge bases are not, and the SESAMO project is an opportunity for developing new knowledge bases or enhance existing ones (like the BDMP knowledge base – see below). KB3 was initially developed for safety analyses, but it is one of the very few tools that has already integrated a security dimension [6]. Its use in the framework of SESAMO will be an opportunity to show its efficiency in other industrial domains than energy.

7.2 SCIENTIFIC CONTRIBUTION

Here is a succinct description of the KB3 workbench, which has no equivalent on the tool market, with references to articles giving further details.

EDF started the development of its KB3 software suite in the 90’s and has been using it since then to perform its dependability studies. This software allows a fast and user-friendly definition of structural and behavioral models by graphical assembly of elementary components described in knowledge bases. Knowledge bases contain generic descriptions of the components typically encountered in a given kind of study. They enable expert knowledge reusability and modularity. They are written in Figaro, an object oriented language also developed by EDF, specifically designed to build stochastic models. Fig. 1 shows the generic principles of the “KB3 workbench”, made of KB3 itself (the model building tool) and various model solvers.
The BDMP (Boolean logic Driven Markov Processes) formalism has been recently adapted from the dependability area [1] to the security domain in order to model attack scenarios. Starting from the theoretical framework of such an adaptation exposed in [2, 3], we have extended the KB3 modeling software platform [4] to let security analysts build and analyze security-oriented BDMP models.

A Specific Knowledge Base for Security. A specific knowledge base has been developed to implement the BDMP formalism for security modeling. It allows easy graphical modeling, interactive simulation and automatic processing for quantification.

This knowledge base was largely inspired by the knowledge base implementing BDMP for dependability analyses which has been used to carry out dozens of studies of real complex systems, particularly in the electrical and nuclear fields.

A Knowledge Base for Security and Safety modeling. Moreover, the two BDMP knowledge bases have been merged into a single one which allows building hybrid models pertaining to both security and safety aspects [5].

Flexibility of the KB3 workbench. Thanks to the FIGARO language, the maintenance and development of knowledge bases requires few resources. Therefore, it will be possible to improve/adapt the existing knowledge bases if it is needed to build models for the use cases of SESAMO. Moreover, it can be envisaged to create a brand new knowledge base if some new ideas appear in the course of the project.

7.3 APPLICATION TO THE SESAMO USE CASES

On the smart grid use case proposed by EDF, two kinds of studies can probably be performed, corresponding to very different amounts of effort.

- A generic modeling of attack scenarios could probably be done using BDMP. This model will not be dedicated to a particular topology of smart grid, but it will produce as results some typical scenarios, described in terms of macroscopic attack steps.
- A precise modeling of a given architecture: this will include both all electrical characteristics of the network and the instrumentation and control, with hypotheses on the kind of links, used protocols… In this case, thanks to Monte Carlo simulation, it should be possible to estimate the probability of undesirable events such as blackouts. But such a model will have to include physical equations, in order to compute for example the load flow in all the situations consecutive to the loss of one or several components (either by accident or because of an attack). For this purpose, it will probably be more efficient to rely on existing simulators of electrical networks than try to introduce the necessary algorithms in the KB3 workbench.

A first feasibility study, taking into account the list of existing simulation tools, must be performed to evaluate whether the building of such a model is possible given the manpower of SESAMO.
Another simple example could be easily studied with BDMP: the case of the remotely controlled medical device. Figure 2 is a first, very simple model, built by EDF on this case. Some events considered as leaves here could be detailed further, and the estimation of times needed to complete the attack steps and of failure rates would be necessary to make this model quantitative and able to produce the most probable scenarios leading to a dangerous situation for the patient.

![Diagram of first safety-security model for a medical device](image)

Figure 4: First safety-security model for a medical device

---

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Electricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
Finally, a third use case with an intermediate (compared to the two previous ones) complexity could be the communication in railways. In this case too, a BDMP could be a good model in order to identify the most probable scenarios (possibly mixing safety and security issues) that could lead to some undesirable event, like a collision of trains.

7.4 REFERENCES


[7] Download address for KB3, the BDMP knowledge bases and examples of models: http://sourceforge.net/projects/visualfigaro/files/ (explore the sub-directories)
8 SAFETY & SECURITY ANALYSIS OF RESILIENT SERVICES IN COMMUNICATION NETWORKS (FTW)

8.1 MOTIVATION.

One challenge for communication infrastructures deployed in safety-critical systems is operating networks over wireless links, compared to wired network links only. This particularly gains importance when transmitting over wireless communication links prone to an extended range of possible attack scenarios.

Starting from traditional approaches to communication in critical systems, future resilient and secure communication services exploit existing communication infrastructures, off-the-shelf equipment, and offer a tight, but yet flexible integration of secure services.

In general, communication networks can be rated by metrics like up-time, responsiveness, and appropriate allocation of resources. Defining appropriate metrics for communication network security is more challenging and often based on empirical data like the number of security events in a certain time period. Given the operation of communication protocol providing services for both safety and security, conflicts may arise. It is essential for network and protocol design to understand those conflicts, and be able to provide a suitable set of selected metrics and means for quantification for subsequent modeling and evaluation purposes. One example is the ability to acknowledge strict temporal bounds formulated as message input requirements for communication, and at the same time provide secure traffic to trusted endpoints in wireless mobile environments.

Further, communication infrastructure will include existing communication networks as part of critical systems. This creates challenges with respect to managing the complexity of heterogeneous network infrastructures and links operated by third party network providers. One approach for satisfying communication properties is to capture the parameterization state space of applied mechanism, and exploit synergies amongst them.

The main motivation is to overcome the current approach of describing, modeling, and evaluating communication networks and protocols in isolation, which considers those tasks for safety-critical systems and secure communication as non-related tasks. The goal is aiming at a joint analysis approach for safe and secure communication architectures, including topologies for authentication, and efficient key distribution and management.

8.2 SCIENTIFIC CONTRIBUTION.

FTW will focus its scientific contribution to the topics of communication networks as listed below. Those are based on modeling mechanism of building blocks described in WP2.

- Extend the means of formulating problem statements using existing tool support, to allow for specifications that reflect the specific needs of heterogeneous wireless communication
networks and accommodate for complex communication overlay networks and consider safety and security requirements.

- Architectural specification of communication infrastructures to accommodate for--and satisfy--identified safety and security properties of communication protocols.
- Enhance current analysis approaches as used for evaluating protocols for safety-critical systems to embrace methods for security. Evaluation is based on
  a) input requirements as probability of message delivery, or and temporal bounds on delay and
  b) networking environment.

This takes into account temporal properties as delay or jitter, resource constraints as bandwidth or coordination, and resource distribution as providing guarantees for resources or applying mechanism for traffic-shaping. It includes embedding the results of identified security threats and attack scenarios, as well as counter-measures, in a quantitative manner.

- Improve methods for validating communication sub-systems within critical infrastructures, to allow modeling of both selected safety and security properties and their interdependencies.
- Address performance of mixed criticality networks, operating services of different quality of service demands and security requirements over the same (logical) network link. The problems here are resource allocation based on observed traffic patterns by different application types, and arbitrating different phases of communication. This requires the consideration of trade-offs with respect to (re-)establishing communication channels, (re-)authentication, key generation, and security contexts.
- Examine methods for stating temporal bounds for worst-case execution times, and apply it to protocols depending on security mechanism, e.g., authentication. Main focus will be on Layer 2 (MAC) protocols and Layer 3/4 (IP/TCP) communication networks.

The main modeling approaches will use stochastic simulation models of communication protocols, and network simulation of network topologies.

Joint analysis approaches of safety and security will utilize BDMP (Boolean logic Driven Markov Processes) formalism for risk assessment of potential weaknesses and threat models, which results will lead to definition of quantifiable metrics of both safety and security. For the analysis of required trade-offs between safety and security like e.g. delay and security level, approaches using Petri Nets are followed.

### 8.3 APPLICATION TO THE SESAMO USE CASES.

The contributions of FTW will cover analysis and assessment of selected building blocks for communication purposes. It will focus on the challenges described in the following use cases within SESAMO:

- Railway Communication (UniControls)
- Smart Grids (EDF)

Means for verification will be covered on a prototyping basis together with the involved SESAMO partners.
8.3.1 Overview of the technical paper “Timed Broadcast via Off-the-Shelf WLAN Distributed Coordination Function for Safety-Critical Systems” included in the Appendix.


Abstract.

Low cost wireless solutions for safety-critical applications are attractive to leverage safety-critical operation in new application areas. This work assesses the feasibility of providing synchronous and time bounded communication to standard IEEE 802.11 devices with low effort modifications. An existing protocol for time bounded communication in wireless systems is adapted to a generic safety-critical application with low bandwidth requirements, but strict bounds on time behavior. Experimental and simulation studies are conducted in which the protocol is implemented on top of IEEE 802.11e Distributed Coordination Function (DCF). The experimental results for packet loss ratio, communication delays, and broadcast completion are used to calibrate a stochastic simulation model that allows to extrapolate the expected long-term performance of the protocol. Both the experimental results and the simulation extrapolation show that necessary availability requirements can be met with 802.11e prioritization in the investigated cross-traffic and interference scenarios.
9  FORMAL METRICS (IIT-CNR)

Leanid Krautsevich (CNR Institute of Informatics and Telematics)
Fabio Martinelli (CNR Institute of Informatics and Telematics)
Artsiom Yautsiukhin (CNR Institute of Informatics and Telematics)

Summary. This section provides a framework for a formal description of security metrics and an advanced analysis of attacker behaviour. The main contribution is the formal model of metrics, which allowed analysing whether the proposed metrics are able to measure security. Moreover, the advanced model of attacker behaviour (which uses some of the proposed security metrics) should provide a more accurate way to analyse strength of a security system.

9.1 MOTIVATION

More than ten years security community has been looking for the metrics which can measure security correctly and unambiguously. A number of different metrics have been proposed from specific ones, which measure a specific part of a system (e.g., time between antivirus updates), to general metrics, which assess security as a whole (e.g., attack surface) [1, 3]. Unfortunately, neither one metric nor a closed set of metrics are widely accepted for correct measurement of security as a whole and currently many of them are used simultaneously.

Such amount and diversity of metrics are caused by our inability to prove that a metric really measures security. For rigid proofs a formal description of metrics is required. Many authors state that there is a need for formal descriptions of security metrics [2, 4]. Such formalism, general for all metrics, must help to define metrics precisely, prove that metrics are the ones we need, and analyse how metrics relate to each other.

We would like to consider what “more secure” relation means. We want to make several steps forward and try to check whether the existing metrics could be used for assessment of security. We propose a formal model for description of existing security metrics and use a simple criterion to check the metrics. Finally, we apply this knowledge to find a more fine-grained model for an attacker, which learns the system step-by-step and adjust his behaviour according to this knowledge.

9.2 SCIENTIFIC CONTRIBUTION

We define a formal model that allows a more accurate discussion about security, security metrics and risk. Using this model we introduce definitions of security, risk and general security metrics.

According to the measurement theory (and representation theorem [8, 9] in particular), there must be a link between empirical evidence (real system behaviour) and objective measurement (i.e., metrics) and the relations defined on both systems. The problem is that there is no a clear, unambiguous, and widely accepted definition of “more secure” relation (on the empirical system). Every inventor of security metrics defines what “more secure” is by means of metrics, but does not prove that the metric really indicates changes in security.

Although, the definition of “more secure” relation is a difficult task, we propose a simple and clear criterion: if all attacks for one system are also relevant for another one, we call the former system ‘more or equally’ secure than the later one.
This Criterion does not allow distinguishing between equal or higher security, thus we call this definition non sensitive. The sensitive criterion can be defined in the following way: if all attacks for one system are also relevant for another one but not otherwise, we call the former system ‘more’ secure than the later one.

We have formally defined the following metrics which can be found in the literature:

1) Number of attacks [12, 10];
2) Minimal cost of attacks [12];
3) Shortest length of attacks;
4) Maximal probability of successful attacks;
5) Risk [6, 7, 5];
6) Average probability of successful attacks;
7) Attack surface metric [13, 14].

9.2.1 Analysis

We have evaluated the formalised metrics against our sensitive and non-sensitive security Criteria. Moreover, we remind that satisfaction of the criteria (sensitive or non-sensitive) does not guarantee that a metric really measures security. We can only say that the metric may be the one which is good for such measurement. In Table 2 “s” means sensitive, “n-s” - non-sensitive, “no” - fails both criteria.

Table 2: Results of satisfaction of security criteria.

<table>
<thead>
<tr>
<th></th>
<th>Number of attacks</th>
<th>Minimal cost</th>
<th>Shortest length</th>
<th>Maximal probability</th>
<th>Risk</th>
<th>Average probability</th>
<th>Attack surface metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>n-s</td>
<td>n-s</td>
<td>n-s</td>
<td>s</td>
<td>s</td>
<td>s</td>
<td>s</td>
</tr>
</tbody>
</table>

9.3 ASSESSMENT OF A SYSTEM WITH AN ADAPTIVE ATTACKER

9.3.1 Motivation

When we consider security strength in a specific context, we should take into account possible behaviour of an attacker. This behaviour determines how malicious external forces are dangerous for the system we consider. In particular, we should know how attackers choose the attacks to follow. This choice depends on the information about the system the attacker possesses and how it collects this knowledge.
Most methods for the analysis of security of computer systems (e.g., networks, Cloud, etc.) consider attackers as omniscient entities which know all weaknesses of a computer system [15, 16]. In addition, attackers are frequently assumed to make only right decisions during an attack and to exploit only the best possible way for the attack.

In contrast, descriptions of real complex attacks (e.g., [17]) show that attackers have limited knowledge of a target system and explore the system step by step during the attack. Attackers make mistakes in their reasoning about the system, and search for alternative ways to compromise the system when the initially selected attack fails. This means that the model of powerful attacker does not provide a real description of a situation, but prepares for a worst case scenario. In reality, security teams have a limited budget and would like to concentrate on the most important security issues that can be solved within a budget.

In this part, we strive for a more refined attacker model introducing the attacker’s view of a system, which is sometimes different from the real system. This view drives the actions of the attacker depending on the knowledge and resources the attacker possesses. Moreover, in our model an attacker may give up on her current attack and follow an alternative attack path. Thus, we get information about alternative attacks (and their probability of occurrence) the attackers will launch against the system. We use Markov Decision Process (MDP) to model the behaviour of attacker as the method for the selection of attack steps.

9.3.2 Scientific contribution

We consider a computer system as an attack graph $G$ that represents the ways to compromise the system [15, 16]. A node $s_i \in S$ of the attack graph denotes a successfully exploited vulnerability and an edge $a_{ij} \in A$ denotes further possible exploitation of vulnerability $s_j$ after previously exploited vulnerability $s_i$. Thus, successful exploitation of vulnerabilities leads an attacker to new states with new privileges. There are several methods for automated construction of attack graphs [16, 18].

We separate the real system and the attacker belief about the system. The attacker’s knowledge about the system determines the set of vulnerabilities that the attacker believes present in the system. The set of vulnerabilities that are believed by the attacker is further reduced according to attacker’s skills and tangible resources. Finally, the attacker has her own view (a graph $G_x = (S_x, A_x)$) of the system.

We assume that the system behaves probabilistically. We introduce probability $P_{r_{ij}}$ of system transition from state $i$ to state $j$ in response to an attacker’s action. For the attacker this probability is:

$$\Pr_{ij} = \Pr_{ij}^\rho \times \Pr_{ij}^{\text{exp}}$$

(20)

where $\Pr_{ij}^\rho$ is the probability that the vulnerability $j$ presents in the systems and $\Pr_{ij}^{\text{exp}}$ is the conditional probability that the vulnerability may be successfully exploited in case it exists in the system. Finally, for all states we assign a reward $r_{ij} \in R$ the attacker gets when reaches the state $s_j$ through executing an action $a_{ij}$.
9.3.3 Attacker’s behaviour

The algorithm for the computation of optimal deterministic policies is the backward induction (see algorithm in Figure 5). The backward induction is algorithm for computation a policy of finite-horizon discrete-time Markov decision problem. Finite horizon means that the number of decision epochs for the attacks is bounded which is true because the attacker never has infinite time for performing the attack. The algorithm finds sets $A^*_t$ of actions that maximise the expected total reward of the attacker. Actions are collected as a set $\Pi$ of policies $\pi$ such that $\pi \in \Pi$. The attacker does not obtain the maximal reward each time. However, in case of several attacks (e.g., within an attack profile) the average reward will be maximal.

\[
t = N \\
\text{for all } s^N \in S \text{ do} \\
\quad u^N(s^N) = r^N(s^N) \\
\text{end for} \\
\text{while } t > 1 \text{ do} \\
\quad t = t - 1 \\
\quad \text{for all } s^t \in S \text{ do} \\
\quad \quad u^t = \max_{a \in A^t} \left\{ r^t(s^t, a^t) + \sum_{a_{ij} \in A_i} Pr^t_{ij} \cdot u^{t+1}(s_j) \right\} \\
\quad \quad A^*_t = \arg \max_{a \in A^t} \left\{ r^t(s^t, a^t) + \sum_{a_{ij} \in A_i} Pr^t_{ij} \cdot u^{t+1}(s_j) \right\} \\
\text{end for} \\
\text{end while}
\]

Figure 5: Algorithm for a deterministic attacker

We modify the behaviour of the deterministic attacker so that she may reconsider her course of action when she cannot complete her current attack path. We assume that the attacker sets $Pr^d_{ij} = 0$ (and $Pr_{ij} = 0$) when she cannot complete an attack step $a_j$ and understands that the vulnerability is absent in the system. In addition, the attacker sets $Pr_{ij} = 0$ for all other edges entering $s_j$ from all states $s_i$. Then, the attacker uses the algorithm from Figure 5 to compute a new strategy using the updated attack graph and the amount of decision epochs left after the initial part of the attack.

The attacker sets $Pr^p_{ij} = 1$ and $Pr_{ij} = Pr_{ij}^{exp}$ if she understands that the vulnerability exists in the system as a result of the unsuccessful attempt of the attack step. Then the attack strategy is recomputed according to Algorithm 1 with the rest of the decision epochs. If the attacker successfully exploits the vulnerability $s_j$ she adds edges $a_{ij}$ and sets $Pr_{ij} = Pr_{ij}$ for all states $s_j$ reachable from $s_i$ in one step. This modification is required to remember the privileges gained by the attacker for future adjustments in her strategy.
We see that for our analysis we need almost the same information required for other methods considering the worst case scenario \[15, 16\]. The algorithm needs the attack graph and sets of probabilities and rewards. As a result the algorithm provides the probabilities of selection one or another branch in the graph, i.e., the probability to execute one or another attack. We also may consider different strategy for selection of alternative path, e.g., an attacker tries several attempts before deciding to give up and try to compromise another vulnerability (see \[19\] for details), but in this case the amount of attempts has to be provided.

9.4 APPLICATION TO THE MEDICAL USE CASE

We consider the Medical Use Case. The hospital saves the data about the health of patients information in an online database service to be accessed by doctors. A competitor medical company would like to compromise the hospital stealing and releasing the sensitive information. The competitor hires professional hackers to attack the server where the database is installed.

The server operates FreeBSD 7 and MySQL 5. The database is managed by an administrator that uses a local workstation operated by Linux Mint 12 with Pidgin Messenger installed. Moreover, the administrator manages the database from her home laptop using a VPN connection to the workstation. The laptop runs Windows 7, Chrome browser, and TUKEVA Password Reminder. The whole system and a possible attack graph are depicted in Figure 6.

![Attack Graph](image)

Figure 6: a) the network system, b) the attack graph of the network system

In our example, the attacker has \(N\) decision epochs and gets terminal rewards ($10K) only if she reaches states 3, 7, 8 i.e. \(r(s_3) = r(s_7) = r(s_8) = 10\) other terminal rewards equal to 0. Instant rewards also equal to 0. For the attack graph presented in Figure 6, the policy is \(\pi = (a_1 = a_{08})\) at the initial state during the first decision epoch. Suppose, the action is unsuccessful because the vulnerability has timely patched by the administrator. The attacker sets the probability \(Pr_{in} = 0\), reconsiders her initial policies using \(N\)-1 decision epochs, and obtains new policy \(\pi = (a'_1 = a_{08})\).

9.5 FUTURE WORK

We would like to improve our model by investigating further towards a probabilistic attacker, the attacker which does not behaves deterministically, and use this knowledge to find the probability of attack selection, required for quantitative assessment of security.
9.6 REFERENCES:


Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikr++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
10 STATE OF THE ART FMEA TECHNIQUES (SAG)

Amandus Kofler (SAG)
Martin Matschnig (SAG)

Summary:

This section deals with FMEA as main analysis technique in safety standards. The main contribution is to show in detail the benefits and methods to guarantee the required safety level. A practical FMEA example on the industrial drives use case will follow in SESAMO WP5.

Motivation

Failure Modes and Effects Analysis (FMEA) is a very common and widely used analysis method for safety critical systems. It is a logical decomposition used to identify and eliminate possible causes of failure. Since it is highly recommended in important safety standards as IEC 61508 and design guidelines like DO 254, it will be a valuable component within the SESAMO context. First FMEA techniques were already introduced in the 1950s and today there are various FMEA tools available. Within the following, the underlying principles and different flavours will be shown and two concrete examples will be presented. One example is based on a traditional flow based on simple charts, while the second one uses a state of the art system level design based approach. A detailed Failure Mode and Criticality Analysis (FMECA) according to IEC 61508 applied to the industrial drives use case, will be done in SESAMO WP5.

Contribution

10.1 FMEA OVERVIEW AND FUNDAMENTALS

Whenever safety relevant hardware is developed, a certification is mandatory. Depending on the applicable safety standard an FMEA is recommended or even mandatory. FMEA is a systematic method for identification and prevention of failures before they occur. The FMEA determines the effect of each failure and identifies single failure points that are critical.

Since its beginning in the 1950s FMEA has evolved from an ad hoc technique, dependent on the designer’s “experience”, to a formal and accepted analysis technique applicable at all phases of the system development process. Failure modes can be described functionally and their effects analyzed early in the design phase. Interface and detailed FMEAs can be applied as the design detail develops. Fault trees can be used to combine the contributions to the failure mode due to design errors, material properties, manufacturing errors, service mistakes, and even user errors to assess the probability with which the failure mode occurs. Failure modes can be eliminated by removing their causes or at least have their probabilities of failure reduced to acceptable levels. [1]
10.2 HISTORY AND RELATION TO STANDARDS

Naval standard Mil-Std-1629 [2], “Procedures for Performing a Failure Mode, Effects and Criticality Analysis” which was initially published in 1974, became the basic approach for doing the analysis.


The most laborious method is described by the Society for Automotive Engineers (SAE) in ARP 926.

FMEA is highly recommended in IEC 61508-3:B4, IEC 61508-7:B.6.6.1 and especially in IEC 61508-2:C.1 it is mandatory as a basis for calculation of the Safe Failure Fraction (SFF).

10.3 CLASSIFICATION AND FLAVORS

As a typical “bottom-up” method FMEA starts with a detailed list of all components within the system. The approach is to identify the failure modes of the individual components and analyze their impacts on the higher levels, climbing up to their effects on the highest system level.

In contrast to this the well known FTA (fault tree analysis) is a ”top-down” method, where an high level error is traced back to the lower system levels.

FTA is also better suitable for analysis. You can calculate the probability of a certain event. Both can be used for risk ranking, i.e., finding out which component of a system contributes to something you are interested in.
Figure 7: FMEA in relation to FTA

FMEA and FTA complement each other. In practice a FTA is performed for larger systems. When a problem is detected within a certain subsystem an FMEA on the smaller subsystem is performed to find out how it behaves.

There are several different types of FMEA targeting different process phases (Concept, Design and Process). Following Table 3: C-FMEA, D-FMEA and P-FMEA[4] and Figure 8: Different FMEA Types[5] give an overview on the main characteristics of those flavors.

<table>
<thead>
<tr>
<th>CONCEPT FMEA (C-FMEA)</th>
<th>DESIGN FMEA (D-FMEA)</th>
<th>PROCESS FMEA (P-FMEA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRIMARY INTENTION</td>
<td>analyze concepts in the early phases before hardware is defined</td>
<td>analyze products and hardware functions before they are released to production</td>
</tr>
<tr>
<td>MAIN FOCUS</td>
<td>potential failure modes associated with the proposed functions of a concept proposal</td>
<td>potential failure modes of products caused by design deficiencies</td>
</tr>
<tr>
<td>HIERARCHY LEVELS</td>
<td>multiple interacting systems individual system subsystem</td>
<td>system subsystem component</td>
</tr>
<tr>
<td>MAJOR BENEFITS</td>
<td>Finding the best concept alternatives, or determine changes to design specifications</td>
<td>Aids in the objective evaluation of design requirements and design alternatives</td>
</tr>
<tr>
<td></td>
<td>Identification of potential failure modes caused by interactions between system components within the concept Increases the probability that failure modes of a proposed concept are considered</td>
<td>Aids in the initial design for manufacturing and assembly requirements Increases the probability that potential failure modes and their effects have been considered in the design/development process Provides additional information to help plan thorough and efficient test programs. Develops a list of potential failure modes ranked according to their effect on the customer. Establishes a priority system for design improvements. Provides an open issue format for recommending and tracking risk reducing actions. Provides future reference to aid in analyzing field concerns.</td>
</tr>
</tbody>
</table>

Table 3: C-FMEA, D-FMEA and P-FMEA[4]
10.4 FAILURE MODE, EFFECTS AND CRITICALITY ANALYSIS

The terms FMEA (failure modes and effects analysis) and FMECA (failure modes, effects, and criticality analysis) are often used almost synonymously. Some publications take the position that FMEA is limited to an analysis of the effects of item failure modes and that FMECA also includes assessing the probability of a failure mode’s occurrence. Others take the view that FMECA extends the analysis to include a ranking of the failure modes based on a combination of their probability of occurrence and the severity of their effects. Within the following the term FMECA is used as the general term that includes prioritizations based on the failure mode’s severity and probability of occurrence.

10.5 GENERAL FMEA APPROACH

The FMECA for HW is a relative simple method, easy to understand and in most cases to implement without problems. There are standard forms and tools available to guarantee a systematic approach. A complete automation will not be possible and would suggest wrong confidence. An experienced team and the help of the certifying institute are needed. The FMEA covers the organized development process. The integration of the certifier and all other stakeholders is an important success factor. The underlying idea of the method is to assign a Risk Priority Number RPN. Operating figures like (P) Probability, (S) Severity and (D) Detection are defined following a catalog.
These numbers are in a range between 1 and 10, so that a RPN results between 1 and 1000. Following Table 4, Table 5 Table 6 show how the numbers can be assigned. The right estimation is the challenge and the greatest weakness of the FMEA approach. The subjective character does not allow a direct comparison of different FMEAs. To prevent wrong figures, in most cases to low numbers, comments or explanations should be given. The greatest advantage of the FMEA compared to other methods is the systematic collecting of experience over failures and there impacts on quality of products or processes. Through documentation the experience and know how will serve other members of the company. FMEA serves to close the quality circle. Further products developments will benefit from the documented know how to avoid failures.

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Effect</th>
<th>SEVERITY of Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Hazardous without warning</td>
<td>Very high severity ranking when a potential failure mode affects safe system operation without warning</td>
</tr>
<tr>
<td>9</td>
<td>Hazardous with warning</td>
<td>Very high severity ranking when a potential failure mode affects safe system operation with warning</td>
</tr>
<tr>
<td>8</td>
<td>Very High</td>
<td>System inoperable with destructive failure without compromising safety</td>
</tr>
<tr>
<td>7</td>
<td>High</td>
<td>System inoperable with equipment damage</td>
</tr>
<tr>
<td>6</td>
<td>Moderate</td>
<td>System inoperable with minor damage</td>
</tr>
<tr>
<td>5</td>
<td>Low</td>
<td>System inoperable without damage</td>
</tr>
<tr>
<td>4</td>
<td>Very Low</td>
<td>System operable with significant degradation of performance</td>
</tr>
<tr>
<td>3</td>
<td>Minor</td>
<td>System operable with some degradation of performance</td>
</tr>
<tr>
<td>2</td>
<td>Very Minor</td>
<td>System operable with minimal interference</td>
</tr>
</tbody>
</table>
### Table 4: Severity of failure ranking

<table>
<thead>
<tr>
<th>Class</th>
<th>Ranking</th>
<th>Category</th>
<th>Hazardous Event Probability</th>
<th>Hazardous Event Frequency Guide</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>Frequent</td>
<td>Likely to occur frequently. The hazard will be continuously experienced</td>
<td>( \geq 10^3/\text{year} )</td>
</tr>
<tr>
<td>B</td>
<td>8 - 9</td>
<td>Probable</td>
<td>Will occur several times. The hazard can be expected to occur often</td>
<td>(&lt; 10^3/\text{year} \geq 1/\text{year} )</td>
</tr>
<tr>
<td>C</td>
<td>6 - 7</td>
<td>Occasional</td>
<td>Likely to occur several times. The hazard can be expected to occur several times</td>
<td>(&lt; 1\text{year} \geq 10^3/\text{year} )</td>
</tr>
<tr>
<td>D</td>
<td>4- 5</td>
<td>Remote</td>
<td>Likely to occur sometime during Project. The hazard can be reasonably expected to occur</td>
<td>(&lt; 10^3/\text{year} \geq 10^4/\text{year} )</td>
</tr>
<tr>
<td>E</td>
<td>2 - 3</td>
<td>Improbable</td>
<td>Unlikely to occur but possible. It can be assumed that the hazard may exceptionally occur</td>
<td>(&lt; 10^4/\text{year} \geq 10^6/\text{year} )</td>
</tr>
<tr>
<td>F</td>
<td>1</td>
<td>Incredible</td>
<td>Extremely unlikely to occur. It can be assumed that the hazard may not occur.</td>
<td>(&lt; 10^6/\text{year} )</td>
</tr>
</tbody>
</table>

### Table 5: Probability of failure ranking

### Table 6: Likelihood of failure detection by design control

---

**Project Partners:**

- Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
### Ranking

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Detection</th>
<th>Likelihood of DETECTION by Design Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Absolute Uncertainty</td>
<td>Design control cannot detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>9</td>
<td>Very Remote</td>
<td>Very remote chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>8</td>
<td>Remote</td>
<td>Remote chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>7</td>
<td>Very Low</td>
<td>Very low chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>6</td>
<td>Low</td>
<td>Low chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>5</td>
<td>Moderate</td>
<td>Moderate chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>4</td>
<td>Moderately High</td>
<td>Moderately High chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>3</td>
<td>High</td>
<td>High chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>2</td>
<td>Very High</td>
<td>Very high chance the design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
<tr>
<td>1</td>
<td>Almost Certain</td>
<td>Design control will detect potential cause/mechanism and subsequent failure mode</td>
</tr>
</tbody>
</table>

The FMEA process splits in the following parts:[5]

- Containment of the system limits
- Structuring of the system
- Define the functions of the structured elements
- Analyze the potential failures, the art of failures and there consequences
- A risk analysis
- Consequences end solution of the prioritized risks
- Tracing of the committed avoidance and recovering activities

Activities:

- reducing the hazard rate
- increase the potential of finding failures
- reevaluate the system thru a FMECA after design changes and after defined process stages
### D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Electricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.

---

<table>
<thead>
<tr>
<th>Potential Failure Mode and Effects Analysis (FMEA)</th>
<th>Risk Priority Number (RPN)</th>
<th>Severity</th>
<th>Probability</th>
<th>Detectability</th>
<th>Recommended Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 10: Exemplary FMEA sheet
10.6 The Safe Failure Fraction SFF

According to IEC 61508-2 Annex C the calculation of the Safe Failure Fraction SFF is mandatory, which has direct influence to the resulting SIL (s. Table 7: SFF and resulting SIL).

<table>
<thead>
<tr>
<th>Safe Failure Fraction (SFF)</th>
<th>Hardware Fault Tolerance (Type A – simple subsystem)</th>
<th>Hardware Fault Tolerance (Type B – complex subsystem)</th>
</tr>
</thead>
<tbody>
<tr>
<td>faults</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>&lt; 60%</td>
<td>SIL1</td>
<td>SIL2</td>
</tr>
<tr>
<td>60%… &lt;90%</td>
<td>SIL2</td>
<td>SIL3</td>
</tr>
<tr>
<td>90%… &lt;99%</td>
<td>SIL3</td>
<td>SIL4</td>
</tr>
<tr>
<td>≥ 99%</td>
<td>SIL3</td>
<td>SIL4</td>
</tr>
</tbody>
</table>

Table 7: SFF and resulting SIL

In practice the SFF calculation often is combined with an FMECA (s. Figure 12: SFF calculation sheet). The FMECA requires the complete HW design, the knowledge of the system behavior and the knowledge of the dangerous states. We also need some timing behavior like the time to repair the faulty system and the time to renew the system. The FMECA works with failure rates in time (FIT), starting with the component list and the FIT rates of known industrial sources like the Siemens SN29500-1 or FIDES 2009 etc. The FMECA delivers absolute values. The SFF calculation like in “EN 61508-2 Annex C” works, when the FIT rates are expected constant over time.

\[ \lambda(t) = \text{failure in time / number of units [1/h]; often } [1/10^9 \text{ h}] = [\text{fit}] \]
Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.

Figure 11: Save Failure Fraction (SFF) as function of failure rates.
### Figure 12: SFF calculation sheet

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Fucntion</th>
<th>Failure impact</th>
<th>% safe</th>
<th>% dang</th>
<th>$\lambda_b$</th>
<th>$\lambda_s$</th>
<th>$\lambda_d$</th>
<th>$\lambda_u$</th>
<th>DCcomp</th>
<th>$\lambda_{du}$</th>
<th>$\lambda_{dd}$</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Resistor</td>
<td>12k R10</td>
<td></td>
<td></td>
<td>0.20</td>
<td>0.25</td>
<td>0.000</td>
<td>0.250</td>
<td>0.000</td>
<td>90</td>
<td>5.0E-3</td>
<td>4.5E-3 will be detected by HW</td>
</tr>
<tr>
<td></td>
<td>Resistor sensor</td>
<td></td>
<td></td>
<td></td>
<td>0.20</td>
<td>0.25</td>
<td>0.000</td>
<td>0.250</td>
<td>0.000</td>
<td>90</td>
<td>5.0E-3</td>
<td>4.5E-3 will be detected by cls.</td>
</tr>
<tr>
<td></td>
<td>constant current</td>
<td></td>
<td></td>
<td></td>
<td>0.20</td>
<td>0.25</td>
<td>0.000</td>
<td>0.250</td>
<td>0.000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Source change to</td>
<td></td>
<td></td>
<td></td>
<td>0.20</td>
<td>0.25</td>
<td>0.000</td>
<td>0.250</td>
<td>0.000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>change to 2R</td>
<td></td>
<td></td>
<td></td>
<td>0.20</td>
<td>0.25</td>
<td>0.000</td>
<td>0.250</td>
<td>0.000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \lambda_s = \lambda_{basis} \times \% \times \text{safe} \]
\[ \lambda_{dd} = \text{DCcomp} \times \lambda_d / 100 \]
\[ \lambda_{du} = (100 - \text{DCcomp}) \times \lambda_d / 100 \]

\[ \text{SFF} = (\sum \lambda_s + \sum \lambda_{dd}) / (\sum \lambda_s + \sum \lambda_{dd} + \sum \lambda_{du}) \]
### Row | Explanation
---|---
**Nr.** | Sequencing number  
**Type** | Component type, all components of the same type follow the same failure model  
**CV** | Component value  
**CI** | Component identifier in the analyzed circuit  
**Function** | Short description of the function of the component  
**\( \lambda_{\text{basic}} \)** | Basic failure rate of the component type  
**Failure** | Type of failure, tests or other revealed mechanism not included  
**Failure impact** | Impact of the failure to the function, tests or other revealed mechanism included  
  | Part of the basic failure rate for the analyzed failure model  
  | Alessandro Birolini: Qualität und Zuverlässigkeit technischer Systeme. Springer, Berlin  
**safe** | 1 when malfunction turns to the safe site  
**dang** | 1 when malfunction turns to the dangerous site  
**\*** | 1 when malfunction has no implication to the overall function, (don't care)
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_s$</td>
<td>calculated failure rate for save states</td>
</tr>
<tr>
<td>$\lambda_d$</td>
<td>calculated failure rate for dangerous states</td>
</tr>
<tr>
<td>$\lambda^*$</td>
<td>calculated failure rate for “don’t care” states</td>
</tr>
<tr>
<td>Findings and containment activities</td>
<td>Description of selve tests or other revealed mechanism, they prevent dangerous failures</td>
</tr>
<tr>
<td>$DC_{comp}$</td>
<td>diagnostic coverage of the above findings and containment activities,</td>
</tr>
<tr>
<td>$\lambda_{du}$</td>
<td>calculated failure rate for undetected dangerous states</td>
</tr>
<tr>
<td>$\lambda_{dd}$</td>
<td>calculated failure rate for detected dangerous states</td>
</tr>
</tbody>
</table>

Figure 13: SFF calculation nomenclature
10.7 Calculation of PFD (Probability of Failure on Demand)

IEC 61508-6 shows detailed information for the quantitative calculations for safety related systems. It gives diagrams and calculations for the PFD value. As well there are tables to calculate the $\beta$ factor, the diagnostic coverage (DC) and safe failure fraction (SFF). We will present the PFD equations for the main systems 1oo1 and 1oo2:

1oo1 - System:

$$PFD = (\lambda_{DU} + \lambda_{DD}) \cdot t_{CE}$$

$$= \lambda_{DU} \left( \frac{T}{2} + MTTR \right) + \lambda_{DD} \cdot MTTR$$

1oo2 - System:

$$PFD = 2 \cdot ((1 - \beta_D)\lambda_{DD} + (1 - \beta)\lambda_{DU})^2 \cdot t_{CE} \cdot t_{GE} + \beta_D \cdot \lambda_{DD} \cdot MTTR + \beta \cdot \lambda_{DU} \left( \frac{T}{2} + MTTR \right)$$

$$t_{CE} = \frac{\lambda_{DU}}{\lambda_D} \left( \frac{T}{2} + MTTR \right) + \frac{\lambda_{DD}}{\lambda_D} \cdot MTTR$$

$$t_{GE} = \frac{\lambda_{DU}}{\lambda_D} \left( \frac{T}{3} + MTTR \right) + \frac{\lambda_{DD}}{\lambda_D} \cdot MTTR$$

The factors in these equations have the following meaning:

$\beta$ The fraction of undetected failures that have a common cause

$\beta_D$ The fraction of detected failures by the diagnostics that have a common cause
MTTR Mean time to restoration (hour)

T Proof-test interval (hour)

T_{CE} Channel equivalent mean down time (hour)

T_{GE} Voted group equivalent mean down time (hour)

SIL requirements according to IEC/EN 61508 for systems operating in low and high demand mode for PFD:

<table>
<thead>
<tr>
<th>Safety integrity level (SIL)</th>
<th>Low demand mode of operation</th>
<th>High demand or continuous mode of operation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PFD (hour)</td>
<td>PFD (hour)</td>
</tr>
<tr>
<td>4</td>
<td>$\geq 10^{-5}$ to $&lt; 10^{-4}$</td>
<td>$\geq 10^{-9}$ to $&lt; 10^{-8}$</td>
</tr>
<tr>
<td>3</td>
<td>$\geq 10^{-4}$ to $&lt; 10^{-3}$</td>
<td>$\geq 10^{-8}$ to $&lt; 10^{-7}$</td>
</tr>
<tr>
<td>2</td>
<td>$\geq 10^{-3}$ to $&lt; 10^{-2}$</td>
<td>$\geq 10^{-7}$ to $&lt; 10^{-6}$</td>
</tr>
<tr>
<td>1</td>
<td>$\geq 10^{-2}$ to $&lt; 10^{-1}$</td>
<td>$\geq 10^{-6}$ to $&lt; 10^{-5}$</td>
</tr>
</tbody>
</table>

10.8 REFERENCES


Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
11 SAFETY AND SECURITY ANALYSIS FOR RAILWAY APPLICATION (UC)

Martin Vitek (Unicontrols)
Stanislav Benes (Unicontrols)
Petr Novobilsky (Unicontrols)

Summary. This contribution gives an overview of safety and security analysis across whole RAMS life cycle of railway application and also provides an overview of safety analysis methods applicable in railway use case with listing their advantages and constraints. This contribution identifies main functions of railway use case (provides secure channel and separation of open and closed transmission system) which will be analysed.

11.1 MOTIVATION.

Currently the development for railway application has to comply with these standards [1][2][3][4]:

1) EN 50126 Railway application – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS). This standard is the railway sector specific application of IEC 61508.
2) EN 50129 Railway application – Communication, signaling and processing systems – Safety-related electronic systems for signalling.
3) EN 50128 Railway application - Communication, signaling and processing systems – Software for railway control and protection systems.
4) EN 50159 Railway application – Communication, signalling and processing systems – Safety related communication in transmission systems.

Standard EN 50126 [1] considers the generic aspects of application’s RAMS life-cycle and define role of risk and safety analysis across whole application life-cycle.

Railway application life-cycle includes following phases (In each phase are defined their safety related parts):

1) Concept.
   a) Review previously achieved safety performance.
   b) Review safety policy and safety goals.
   c) Consider safety implication of application.
2) System definition and operational context.
   a) Establish overall safety plan.
   b) Identify influence on safety of existing infrastructure constraints.
   c) Evaluate past experience.
   d) Perform preliminary risk analysis.
3) Risk analysis and evaluation.
   a) Determine risk acceptance principles and criteria.
   b) Perform risk analysis.
   c) Set-up hazard log.
   d) Perform risk evaluation.
4) Specification of system requirement.
   a) Specify safety requirement.
   • Safety requirement may be categorized by follows:
- Technical safety requirements. Technical safety requirements do not derive from the system functions, but from their technical implementation.
- Operational safety requirements. Operational safety requirements comprise e.g. expected operational procedures for normal and abnormal operation modes, assumptions about safety-related operation restriction etc.

![Figure 14: Categorization safety requirements](image)

b) Define safety acceptance criteria.
c) Define safety validation plan.

5) Architecture and apportionment of system requirements.
a) Specify sub-system and component safety requirements.
b) Define sub-system and component safety acceptance criteria.
c) Update safety plan and safety validation plan (if necessary).

6) Design and implementation.
a) Implement safety plan by review, analysis, testing and data assessment of hazard log, hazard analysis and justify safety-related (sub-)system design decision.
b) Prepare generic safety case.
c) Assess generic safety case.
d) Undertake program control of safety management.
e) Undertake program control of sub-contractors and suppliers control.

7) Manufacture.
a) Implement safety plan by review, analysis, testing and data assessment.
b) Use hazard log.

8) Integration.
a) Establish installation program.
b) Implement installation program.

9) System validation.
a) Establish program of commissioning.

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
b) Implement program of commissioning.

c) Implement validation plan.

d) Prepare specific application safety case.

10) System acceptance.

a) Assess specific application safety case.

11) Operation, maintenance and performance.

a) Undertake on going maintenance focusing on safety.

b) Undertake on going monitoring focusing on safety.

c) Undertake on going hazard log.

12) Performance monitoring.

a) Collect, analyze, evaluate and use performance and safety statistic.

13) Modification and regeneration.

a) Consider the safety impacts on modification and regeneration.

14) Decommissioning.

a) Establish safety plan.

b) Perform hazard analysis and risk assessment.

c) Implement safety plan.

Set of above standards [1][2][3] doesn’t contain any security aspect (explicitly defined), but in these standards are not explicitly defined if hazard can be caused accidentally (could hide security aspects). There is one exception, standard EN 50159 [4] considers the use of cryptography techniques in relation to communication protocols in open transmission system (concern to protection against masqueraded messages), but doesn’t consider common security aspects of IT. Our opinion is consistent with the claim of Robert Stroud in D1.1 Functional safety and security requirements_v01 chapter A.5.3 Gaps in existing standards with respect to safety and security.

For security management are usable following standards:


Now railway community eagerly awaits for new set of standards, that will replace above set of standards:


Application of these standards for railway use case should aim to creation methodology of railway application verification, considering both safety and security aspects.

11.2 SCIENTIFIC CONTRIBUTION.

11.2.1 Safety analysis and evaluation in railway industry

The standard IEC 60300-3-1 (Application guide – Analysis techniques for dependability – Guide of methodology) [5] provides a comprehensive overview of safety/hazard analysis techniques in conjunction with their usability.

Overview Safety/Risk analysis method which can be used for railway application development are listed in following table.

<table>
<thead>
<tr>
<th>Technique/Method Ref. to standard</th>
<th>Hazard identification</th>
<th>Qualitative analysis</th>
<th>Quantitative analysis</th>
<th>Safety demonstration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hazard and Operational Analysis (HAZOP) IEC 61882</td>
<td>YES.</td>
<td>Identification causes and consequences.</td>
<td>NO</td>
<td>Partially useful.</td>
</tr>
<tr>
<td>Rapid Ranking Analysis (RRA)</td>
<td>YES (partially)</td>
<td>Useful for preliminary hazard analysis and for identifying and ranking hazards for following detailed analysis.</td>
<td>NO</td>
<td>Possible for recording rationale for not performing detailed analysis of low ranking hazards.</td>
</tr>
<tr>
<td>Failure Mode, Effects and Critically Analysis (FMECA) EN 60812 [6]</td>
<td>YES</td>
<td>Identification consequences of failures. Useful for parallel structures in addition to ETA.</td>
<td>Calculation of failure rate and their severity.</td>
<td>Useful for single and parallel structures in addition to FTA for causal analysis.</td>
</tr>
<tr>
<td>Event Tree Analysis (ETA) EN 62502</td>
<td>NO</td>
<td>Sequence of failures.</td>
<td>Calculation of failure rate.</td>
<td>Useful to visualize consequences of system failure.</td>
</tr>
<tr>
<td>Fault Tree Analysis (FTA) EN 61025</td>
<td>NO</td>
<td>Combination of failure states and theirs consequences.</td>
<td>Calculation of reliability and availability parameters.</td>
<td>Useful for multiple structures, causal analysis.</td>
</tr>
<tr>
<td>Common Cause Analysis (CCA)</td>
<td>NO</td>
<td>-</td>
<td>NO</td>
<td>Often used with a FMECA and also needed to justify AND-gates in FTA.</td>
</tr>
<tr>
<td>Deductive Cause-Consequence Analysis (DCCA) *</td>
<td>YES</td>
<td>Identification consequences of failures and sequence of failures.</td>
<td>Calculation of reliability and availability parameters.</td>
<td>DCCA is a formal generalization of FMECA and FTA useful for multiple structures, causal analysis.</td>
</tr>
<tr>
<td>Markov techniques EN 61165</td>
<td>NO</td>
<td>Sequence of failures.</td>
<td>Calculation of reliability and availability parameters.</td>
<td>Useful especially for modelling states and fault sequences (especially in case when static FTA isn’t applicable).</td>
</tr>
<tr>
<td>Petri net</td>
<td>NO</td>
<td>Sequence of failures.</td>
<td>Useful for providing system description for Markov techniques.</td>
<td>Useful especially for modelling states and fault sequences.</td>
</tr>
<tr>
<td>Reliability Block Diagram (RBD) EN 61078</td>
<td>Useful as a support for HAZOP.</td>
<td>Paths to attach of workable state.</td>
<td>Calculation of reliability and availability parameters.</td>
<td>Useful for systems, which can be divided into independent block (physical, software, functional etc.).</td>
</tr>
</tbody>
</table>

* Method isn’t contained in standard IEC 60300-3-1.
Table 1: Overview of safety/risk analysis method

The safety/hazard analysis methods can be divided into two groups:

1) Deductive method (top-down, reasoning from general to specific). These methods are based on knowledge of failures of base element and functional structure of system. Defines relations between base element failures, system failures, malfunctions, operational constraints and system integrity degradation. The base is identification of top level event. Next step is identification of events causing top level event/state. Analyse is repeated recursively for each identified event/state. Recursion is performing until base event/state is found. The base event can’t be further developed onto low level events. In other words, some specific system state was identified and using deductive method chains of basic system states contributing to this state can be built up. Deductive methods can be applied to identification path (logical chain) how the system state can occur. This group includes e.g. FTA, RBD.

2) Inductive method (bottom-up, reasoning from specific cases to general conclusion). These methods are based on identification of component failures. For each kind of failure is derived consequence for next higher component level. In other words set of states of system components were identified, using inductive method can be determined their effect to overall system. By iteration failure consequences are identified up to level of overall system. This group includes e.g. FMEA/FMECA, ETA, Markov techniques.

FMEA

FMEA (Failure Modes – identification failure modes of all system components, Effect Analysis – studying consequences of these failures on state of upper system levels) [6][7][8] is common and widely used method for safety/risk analysis, belonging into group of inductive method. FMEA can be described as systematic tabular technique that explores path in which system components can fail and assesses the consequences of these fails. This method is relevant to use when new system is developed, changes of system or operating condition are made.

It concerns to identification failure states of all base system components (or as many as possible), evaluate their influences/consequences to state of higher level of system (repeating this up to level of overall system) and identification actions to mitigate consequences of failures.

The main advantages of FMEA are following:

1) Systematic investigation of relation between cause and consequences of failure states.
2) Provide preliminary information about probability of critical failure modes, especially single point failure modes which are able to propagate to higher system levels.
3) Investigation of consequences derived from specific causes or initial events.
4) Method is able identify root causes of consequences on system level.
5) Provide base framework of definition measures for risk mitigation.
6) Method is useful for preliminary analysis of new systems.
7) Provide data for trade-off studies.
8) Provide historical documentation for future reference to aid in analysis for design changes.

The main constraints of FMEA are following:

1) Result data should be extensive even for relatively simple system.
2) Method should be complex or unusable if direct relation between cause and consequences doesn’t exist.
3) Method should be complex investigating time sequences of failure modes.
4) Method isn’t able to discover complex failure modes.

FMEA is carried out in following steps:

1) Balance that is useful to use FMEA.
2) Determination boundaries of system for analysis.
3) Understanding system requirements and function.
4) Define failure and success criteria.
5) Creation of system block diagram.
6) Identification of failure states of system items and their consequences on overall system.
7) Summarise each failure effect.
8) Report creation.

All result of FMEA analysis is recorded in FMEA specific table. Each record usually contains component description, failure mode description, consequences on higher system level, consequences on overall system, severity, occurrence, detection.

FMECA

FMECA (Failure Modes Effects and Critically Analysis) is extension of FMEA, which adds assessing the probability of failure’s occurrence and classification of each failure consequences according to their severity and probability of occurrence.

FTA
FTA (Fault Tree Analysis) is now commonly applied method for safety/risk analysis, belonging into group of deductive methods. This method can provide qualitative and quantitative results. Result of qualitative analysis is identification of causes (failure states, or their combination), which lead to undesirable system state (top event). Result of quantitative analysis is probability of undesirable system state.

FTA describes failure behaviour of system resulting in top event of fault tree. The fault tree is a logical model of the various event combinations that will result in the occurrence of the top event (undesirable system state). The fault tree structure starts on top event (top level of system) and progresses in branches developing its causes in lower level of system, until basic level of system is attached (appearing of basic or undeveloped events). In fault tree each event is connected to its causes by a gate, describing mechanism how causes leads to event. Basis event is event on bottom of fault tree, it can’t be developed to its causes. Base event contribution to occurrence of top event is expressed in logical cut. Logical cut is combination of basis events leading to top event.

It is important to understand, that fault tree isn’t model of whole failure behaviour of system, but it corresponds only for specific undesirable event, which is in fault tree represented as top event.

The symbols used for graphical presentation of fault tree are grouped:

1) Event symbols.
2) Gate symbols.
3) Transfer symbols.

The main advantages of FTA are following:

1) FTA should be started in early developing phase and should be develop in later developing phases.
2) FTA can systematically identify and describe logical path leading from top event to base causes.
3) FTA provides graphical display of events chains leading to the top event.
4) FTA provides transformation from logical model to evaluation of appropriate probability indicators.
5) FTA can describe complex influences between system components (e.g. influences hardware-software) leading to top event.
6) Documentation of analytical report.

The main constraints of FTA are following:

1) FTA can’t correct resolve time or sequential event dependencies.
2) FTA has constraints regard to system reconfiguration or time dependent behaviour of system.

These constraints could be resolved by combination FTA with Markov techniques. This combination extended set of static gates with new dynamic gates (e.g. SPARE, priority AND - PAND, sequential gate - SEQ). Functionality of dynamic gate is modelled by Markov chains.
FTA is carried out by following steps:

1) Definition of undesired event (top event).
2) Understanding of system.
3) Definition system boundary.
4) Construction of fault tree.
5) Evaluation of fault tree.
6) Creation of report.

**Combination of FMEA and FTA**

In development of railway applications is recommended to use combination of FMEA and FTA. Benefits of combined analysis are:

1) The using of these two method, complementing each other (FMEA is inductive method, FTA is deductive method), is regarded as an argument for increasing credibility of analysis.
2) Safety standards often requires analysis of both single failure and failure combination. FMEA is suitable for analysis of single failure and FTA is suitable for the both requirements.
3) FMEA is useful for full identification of single failures and investigation of their consequences on system. It isn’t able investigate failure combination and their consequences at a system level. FTA is useful method for causal analysis of undesirable events, it can reveal event chains (failure combination) leading to undesirable system state. It isn’t useful for finding all possible failures.

Analysis consistency could be verified using simple rules:

1) Any failure identified in FMEA having consequences on overall system, must exist as single point event (in minimal critical cut) in FTA.
2) Any single point event in FTA, must be identified in FMEA.

**RBD**

RBD(Reliability Block Diagram) [12] model is widely used due its simplicity. RBD is graphical method for modelling how component of system contributes to the success/failure of overall system. Diagram defines logical interaction of failures in system which is decomposed on subsystems or components, that are required to success operation of overall system.

---

**Project Partners:**
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
There are several important assumptions for construction RBD:

1) Each block can be only bimodal (operate/failure).
2) Failures of blocks are statistically independent.
3) RBD must correspond to the definition of what constitutes system failure.
4) The reliability of each individual block must be known.

If these assumptions are met, it is possible to calculate reliability of overall system.

RBD diagram consists of block representing system components or functions and oriented connectors representing part of logical path leading to success of overall system. Each diagram starts on the left side on input node and ends on the right side on output node. Successful system state requires one or more path through RBD (from input node to output node). The Boolean algebra expressions are used to describe paths leading to successful system operating or combination of failures that cause failure of overall system.

It is important to understand, that for system description should be required more than one RBD. This is true in system which is capable performing several functions.

The main advantages of RBD are following:

1) RBD can be often directly derived from functional diagram of system.
2) RBD can be used for most types of system configuration (e.g. serial, parallel, serial-parallel, parallel-serial, passive redundancy, m out of n redundancy …) and their combination. For complex RBD can be used method of simplifying RBD.
3) RBD shows failure logic of the system with the block diagram.
4) RBD can be used to identify critical components and to show how the system responds to particular failure mode.

The main constraints of RBD are following:

1) RBD can’t provide causal identification of failure states (identification of cause-consequence chains).
2) RBD requires probability model for each block in diagram.
3) Analysis is generally limited to un-repairable systems.
4) RBD can’t be used in cases where it is important to consider failure sequences or time dependencies.

ETA
ETA (Event Tree Analysis) [13] [14] is inductive (bottom-up) techniques used for identifying and evaluating the possible sequences of events in a potential accident scenario following the occurrence of failure or undesired event that starts the considered accident scenario (initialling event). The accident scenario is a set of events beginning in initialling event, flowing through branch event and leading to the undesired end state (outcome). ETA uses logical tree structure named event tree. The goal of ETA is determinate condition under which initialling event lead to serious result or if the event is successfully controlled by the safety systems and procedures implemented in the system design, identify and evaluate all of possible consequence chains after an initial event occurs. ETA provides capability to evaluate probability for each outcome arising from assessed initialling event.

The base assumptions for construction of event tree are following:

1) Failure probabilities in branch events are independents or strong dependent. Strong dependence is resolved by sequention of events with strong dependence [15].
2) Distinct path from branch events must be mutually exclusive.
3) For using Boolean method, there is a condition, each branch event must be divided into binary.

Event tree is a model of accident scenario starting by occurrence of initialling events and progressing through scenario via branch events to the final state. The branch events are events that are able to mitigate, or aggravate the accident scenario (e.g. function of safety systems, alarm warnings, influence of environment, operator intervention ….). The probability of branch event can be obtained from FTA of the branch event.

ETA involves following steps:

1) Define system, system boundaries, subsystems and interfaces.
2) Identify initial events.
3) Create accident scenario.
4) Identify branch events in accident scenario.
5) Construct event tree.
6) Determine probabilities for branch events (e.g. using FTA).
7) Identify outcome risk.
8) Calculate probabilities for outcome risk (using Boolean logic).
9) Recommend corrective action for mitigate risk.
10) Document ETA.

The main advantages of ETA are following:

---

**Project Partners:**
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
1) ETA is usable for wide spectrum of system types.
2) ETA provides visual displaying cause-consequences chains following after initial events.
3) ETA can model sequences and interaction of mitigation factors, which follow after initial events.
4) ETA can combines hardware, software, environment and human interaction.
5) ETA can identify potential single point failures.

The main constraints of ETA are following:

1) ETA can’t identify initial events. This is task of the analyst.
2) ETA can overlook hidden dependencies in system.
3) Partial success/failure is not distinguishable.
4) ETA can only evaluate systems with time invariant probability or failure rate.

When combining event tree and fault tree it is possible to choose between two approaches [16]:

1) SELF/FTL(Small Event Trees – Large Fault Trees/Fault Tree Linking) approach uses relatively small events trees to represent the combination of functional(brand) events, which follows initial events. Fault tree models are used to represent the combinations of basic event failures that would result in the failure of each functional (branch) event in the event tree (Fault tree is used as model of behavior of branch event).
2) LESF/ETL(Large Event Trees – Small Fault Trees/Event Tree Linking) approach relatively large event tree involving states of all subsystem which support system. The fault trees associated with functional (branch) event are developed like that don’t share any basics event (all functional events must be mutually independent). If some fault trees share some basic event, this basic event is moved into event tree as new functional event.

**Markov techniques**

Markov techniques [17] [19] are suitable for modelling and analysing of dynamic system, distributed or fault tolerant systems that can be difficult, or even impossible to model with classical techniques like as FTA, FMEA, or RBD.

Markov techniques are based on Markov process, which is stochastic process [18] whose behaviour is depend only on upon of current state of the system. The sequence of step by which the system entered to the current state is irrelevant for future behaviour.

In formal terms is Markov property for discrete process is that:

\[
P (X_{n+1} = j | X_0 = i_0, X_1=i_1 \ldots X_n=i_n) = P (X_{n+1} = j | X_n = i) \quad (1)
\]

In formal terms is Markov property for continuous process is that:
D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

A Markov process may be described as homogenous, or non-homogenous. Homogenous Markov process is characterized by constant transition rates between the states.

For safety/reliability analysis is widely used Homogenous Markov Process.

A set of first-order differential equation describing the probability of being in each state for each time interval. The number of equations is equal the number of system states.

\[
P' = [A]P
\]  

(3)

Where \( P \) is vector of probabilities of all states and \( A \) is matrix of transaction rates. Markov process is fully defined if transition matrix and the initial probability vector is known.

Markov modelling technique uses state diagram, which consist only two symbols:

1) Circle – representing state of system.
2) Arrows – representing transition between two states. Each transition must be evaluated by transition rate. The system can transit from one to another state whenever a failure or repair occurs.

The state diagram identifies all the discrete states and the possible transitions between those states. The diagram with evaluated transitions is essentially a graphical representation of set of first-order differential equation describing system (3).

Markov analysis involves following steps:

1) Definition of the state space of system.
2) Allocation of transition rates between states.
3) Definition of output indicators.
4) Generating of mathematical model (matrix of transition rates) and its processing by appropriate SW.

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
5) Result interpretation.

The main advantages of Markov techniques are following:

1) Techniques provide flexible model for analysis of system behaviour.
2) Techniques can model redundant configuration, complex maintenance policy, complex models of treatment of failure modes, degraded states of operation and common cause failures.
3) Techniques can model sequences of events.
4) Techniques provide probability solutions for modules, that are to be integrated into another models e.g. FTA.

The main constraints of Markov techniques are following:

1) With the increasing number of system components, exponentially growing number of states in Markov model. These constraints can be partially resolved using Markov Model Reduction Techniques.
2) Standard solution of Markov model doesn’t provide specific indicators e.g. MTTF (Mean Time To Failure), or MTTR (Mean Time To Repair).

**DFTA**

DFTA (Dynamic Fault Tree Analysis) is a combination of FTA and Markov techniques, which resolves some constraints of separate using these methods (for FTA (1,2), for Markov techniques (1)). This method involves new types of gates, which represented some Markov models. These models are evaluated separately and are hidden from the user as gates. Outputs from these models are inputs for FTA.

Examples of new gates are following:

1) **PRIORITY AND.** The output occurs if the inputs occur in a specific sequence, which is specified by conditioning events (two inputs).
2) **SEQUENTIAL** The output occurs if the inputs occur in a specific sequence, which is specified by conditioning events (two or more inputs).
3) **SPARE.** The output occurs if the number of functional spare/backup components is less then required.

More powerful alternative to DFTA is **BDMP** (Boolean logic Driven Markov Processes) method [20].
11.2.2 Security analysis and evaluation in railway industry

Standards [1][2][3] for development of railway applications doesn’t contain explicitly defined security aspects.

If we assume that, safety and security are common properties of railway application (hazard can be caused unintentionally or deliberately), we should include security analysis into Phase 3 of RAMS life-cycle (Risk analysis and evaluation) of railway application (Of course it is first approximation, would be better recognise security aspects whole application RAMS life-cycles, but it is comprehensive comparative study of safety and security standards).

The results of security analysis performed in Phase 3 will be used in following phases of life-cycle:

1) Phase 4 - Specification of system requirement.
2) Phase 5 - Architecture and apportionment of system requirements.
3) Phase 6 - Design and implementation.
4) Phase 9 - System validation.

It isn’t enough recognise, analyse and evaluate hazards (safety aspect) and treats (security aspect) separately, but it is necessary found and evaluate influence between them (especially if this influence is inverse relationship). To simplify this task is appropriate using of RBD (see Table 1) method for dividing application into independent blocks and perform safety and security analysis for these blocks and assess influences between safety and security requirements for these blocks.

11.3 APPLICATION TO THE SESAMO USE CASES.

We intend to use subset of described safety analysis method, or their combinations in our use case ESSI (Embedded Safety and Security Interface). ESSI is intended as device which provide safety and security communication among applications in distributed control system through Open Transmission System.
Figure 15: Generic use case of ESSI

Figure 16: Application specific use case of ESSI

ESSI has following main functions:

1) The creation of secure communication channel through Open Transmission System. Creation of secure channel is based on inclusion of security layer between transport layer and application layer of the ISO/OSI stack. This security layer should be transparent for communication of safety related applications positioned in Closed Transmission Systems.
2) The separation of Closed Transmission System (e.g. dedicated network for safety-critical applications) from Open Transmission System. This separation must block propagation of security attack from Open Transmission System to Closed Transmission System. Using of partitioning is intended for satisfy of this requirement.

3) Multiplexing and de-multiplexing of communication messages of safety relevant applications.

These main functions will be object of safety and security analysis. For first main function the goal is improve security properties of secure channel by improving integrity and authenticity protection mechanism based on result of security analysis. This includes analysis of possible protective codes (e.g. MAC, HMAC, digital signature) and authentication mechanism between pair of ESSI providing security channel. Authentication mechanism involves establishing and releasing of secure channel. Improving of security can influence to safety by adding communication delay. For second main function the main goal is improve security properties of separation of transmission systems. For third main function we suppose only safety relevant aspect.

Closed Transmission System must meet following the condition (in accordance with EN 50159 [4]):

1) List of devices, which are able connect to transmission system, has to be known and invariable (Precisely defined network traffic). Configuration of transmission system has to be included into safety case.
2) Properties of transmission have to be known and stable.
3) Risk of unauthorised access must be inappreciable.

If transmission system (network) does not meet at least one of this conditions, let as have it as open environment (open transmission system, e.g. Internet, 802.11 or common technological site, which may be shared by several control systems etc.).

Following figure contain simple block schema of ESSI, which define composition of ESSI and relation of ESSI components to safety and security analysis.
Figure 17: Block schema of ESSI with relation to safety/security analysis

The inputs for safety and security analysis are high level requirements listed in WP1 (D1.2 – Use Case Specification_v01).

Following table contain preview of high level requirements, and their relation to top events (Top event specification is a part of security, and safety analysis).

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Requirement name</th>
<th>Top event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Railway 5.6.0</td>
<td>User data must be protected from modification during transfer via open transmission environment.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 5.7.0</td>
<td>Mechanism of ESSI identification signs assignment must be protected from an unauthorised access via service interface and ambiguity.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 5.9.0</td>
<td>Configuration data must be protected from unauthorised change via service interface.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 5.10.0</td>
<td>The ESSI SW image must be protected from unauthorised change.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 1.1.0</td>
<td>The ESSI must provide possibility of communication redundancy.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 1.5.0</td>
<td>Data multiplexing must be done according to their safety significance.</td>
<td>Will be added after analysis.</td>
</tr>
</tbody>
</table>
Page 5

D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Railway 1.2.0</td>
<td>The development must meet the requirements of EN 50129 and EN 50128 for SIL system.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 1.3.0</td>
<td>The maximal data delay of data transfer caused by asymmetric cryptographic algorithm is 20s.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>Railway 1.4.0</td>
<td>Operating system certifiable according to EN 50128 SIL 4 by supporting space partitioning.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>*</td>
<td>Operating system certifiable according to EN 50128 SIL 4 by supporting time partitioning.</td>
<td>Will be added after analysis.</td>
</tr>
<tr>
<td>*</td>
<td>ESSI must be open only for data from closed transmission system and from ESSI secure channels.</td>
<td>Will be added after analysis.</td>
</tr>
</tbody>
</table>

* New high level requirement for next revision D1.1

Table 2: Overview of High level requirement

11.4 REFERENCES


Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.


[12] EN 61078 Analysis techniques for dependability – Reliability block diagram
    and Boolean methods.

[13] EN 62502 Analysis techniques for dependability – Event tree analysis


[16] Olivier Nusbaumer, Antonie Rauzy: Fault Tree Linking versus Event Tree
    Linking Approaches: A Mathematical And Algorithmic Reconciliation.

[17] EN 61165 Application of Markov techniques

    Academic Press

    Instrument Society of America.

    advantages of fault-trees and Markov models: Boolean logic driven Markov
    processes.
SAFETY ANALYSIS (IFAG AND ESY)

Heidt Laurent (IFAG)
Badstuebner Frank (IFAG)
Griessing Alexander (IFAG)
Spratler Christian (ESY)
Bialek Jan (ESY)

Summary

This section gives a detailed description of state of the art safety analysis methods use in automotive domain to comply with the functional safety standard ISO26262.

This allows a better understanding of the type of analysis required in automotive domain and enables a future integration with or extension to security analysis.

11.5 MOTIVATION

The product development for automotive has to comply with the ISO 26262 standard “Read vehicles – functional safety” (first released edition from November 2011). The standard is using a safety goal approach with a related Automotive Safety Integrity Level to define across the whole product development lifecycle the measures and methods to be applied. The standard is using 3 main V cycles to structure the development of a product: system, hardware and software level.

The two analysis methods described in this chapter have to be performed to confirm that the software architecture and the defined safety mechanisms are appropriate and exhaustive to detect, to contain or to correct all failure modes that could lead to the violation of a safety goal.

The work product generated with such an activities have to be made available to the assessor of the product.

11.6 DESCRIPTION OF SAFETY CRITICALITY ANALYSIS (SCA)

The analysis method described below has to be performed during the software architecture phase (ISO26262 part 6 clause 7). As stated in the ISO 26262, the aim is to:

- identify or confirm the safety-related parts of the software; and
- support the specification and verify the efficiency of the safety mechanisms

The analysis has to be performed according to the description in ISO26262 part 9, clause 8. The Safety Criticality Analysis (SCA) and HAZOP (Hazard and Operability) Analysis are qualitative methods that can be used to verify the Software Architecture and SW Safety Concept.

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
This type of analysis is done on software module level once the software architecture is released.

### 11.6.1 SCA Prerequisites

To perform a safety criticality analysis on module level, following documents need at least to be available:

- Software safety requirements (resulting from ISO26262 part 6, 6.5.1)
- Software architectural specification (resulting from ISO26262 part 6, 7.5.1)

![Figure 18: SCA within V-Cycle](image)

**11.6.1.1 Software safety requirements**

The software safety requirements shall address each software-based function whose failure could lead to a violation of a technical safety requirement allocated to software. The concept and system development phases that leads to the definition of the technical safety requirements is described in ISO26262 part 3 (concept phase) and part 4 (system development). The structure can be summarized with the following figure:
Important is that each type of safety requirement (functional, technical, software) always has following properties:

- An assigned ASIL
- An allocated sub-system or software module

11.6.1.2 Software architectural specification

The aim of the software architectural specification is to represent all software components and their interactions in a hierarchical structure. Static aspects as well as dynamic aspects are described. In order to develop a software architectural specification both software safety requirements as well as all non-safety-related requirements are implemented. Hence in this sub-phase safety-related and non-safety-related requirements are handled within one development process.

Following ISO26262 requirements have to be considered during the architectural design:

- “Freedom from interference between software elements” (part 6 Annex D),
- “Criteria for coexistence of elements” (part 9-6)
- “Requirements decomposition with respect to ASIL tailoring” (part 9-5), in short “ASIL decomposition”

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
11.6.2 Identification of Software structure

Safety Criticality Analysis should be performed separately for each SW module that has at least one ASIL requirement (i.e. not QM). To achieve the necessary level of detail (i.e. during investigation of failure modes and mapping to appropriate safety measures) the analysis is performed on API or SW function level. However, to reduce effort, it is recommended to group several APIs with similar functionality and failure modes into SW-function groups that are analyzed together.

11.6.2.1 Define SW function groups

The first step is to list all APIs or SW functions of the SW module, together with a short description of its functionality. Then, group APIs with similar functionality and failure modes into SW-function groups.

For example, it may be possible to define groups like:

- Configuration group: a group of function that configures how the software module will process its main functionality
- Control group: a group of function that enable or disable the software module functionalities

Finally, document the parent for each entry (i.e. the element that is one level up in the hierarchy tree: SW module - SW function group - SW function)

11.6.2.2 Confirm the ASIL of safety related part

An ASIL (ASIL-A to ASIL-D) must be allocated to each function. Typically all SW functions inherit the same ASIL as the SW module in a top-down approach. There may be reason why some functions or a group of function could have a lower ASIL than the main function inside a module, because they are not participating to a safety goal. It is not recommended to assigning different (e.g. lower) ASIL targets to single SW functions, because this may require the implementation of additional measures to meet the criteria for coexistence (ISO26262 part 9, clause 6).

Each SW function is assigned a criticality class (C0 to C3)

- C1: Interference free. No interference with safety related functionality
- C2: Safety relevant. Latent fault
- C3: Safety critical. Single point fault

The assigned criticality is used to confirm the ASIL requirement bottom-up for each function. In case that the assigned criticality suggests a lower ASIL than the one inherited from the SW module, it may be better still not to reduce the ASIL target for that SW function, because this would require the implementation of additional measures to meet the criteria for coexistence.

The combination of ASIL target and criticality defines for each SW function what kind of safety measures should be considered for implementation for that SW function with reference to ISO 26262-6: Table 4 or similar.

For example, if an ASIL D software module contains a function that is not safety related (i.e. does not have any safety requirement), then this function shall only implement measures to ensure independence and interference-freeness.
11.6.3 Safety Criticality Analysis

The Safety Criticality Analysis is a verification method to confirm that all possible failure modes are covered by appropriate safety measures (as specified in the software technical safety concept). It is conducted in the following steps:

11.6.3.1 Search for possible failure modes

Search for possible failure modes in the different SW functions / SW function groups. Please consider during the searching process following type of possible errors:

- Non-fulfillment of the required functionality (i.e. requirement negation)
- Corruption of internal data structures or API parameters of a SW function (e.g. data corruption, invalid parameter combinations …)
- Corruption of the program flow within a SW function (e.g. unintended program flow …)
- Disturbance of the timing of a SW function (e.g. interruption, waiting for resources, dead-lock …)
- Other known or unknown failure scenarios

11.6.3.2 Documentation of possible root causes for the failure mode

For each type of possible failure mode, document all possible root causes. If a root cause cannot be found, the failure mode may still be considered plausible based on a conservative safety-oriented expert judgment.

Identifying the root causes may help to define a proper safety measures that is preventing the root cause to happen instead of detecting the failure. This is important since preventing root cause to happen would prevent the system to go to the safe state and hence not reduce the availability of the system.

11.6.3.3 Mapping of failure mode to existing safety measure

For each failure mode:

- Identify an effective safety measure to prevent, detect and/or control the failure mode and its effects
- Confirm that the safety measure meets the recommendations of ISO 26262-6 Table 4 with respect to the ASIL target of the related SW functions or SW function group
- Confirm that the safety measure is already defined in the Software architectural specification (for requirement traceability)

Note: If a failure mode is identified that can affect SW modules of criticality C2 or C3, and that is not prevented, detected and/or controlled by an appropriate set of defined safety measures, then...
additional safety measures must be specified in the software architectural specification. A change management process shall be used for such update.

11.6.4 HAZOP Analysis
The HAZOP Analysis is a more systematic and thorough analysis method, that can be performed if the regular Safety Criticality Analysis still leaves any doubt (e.g. regarding completeness or detail), or if a second independent verification / analysis is desired. Therefore an HAZOP analysis can be done for the complete SW module or for specific SW functions. The HAZOP Analysis is performed in 2 steps.

11.6.4.1 Systematic keyword
The HAZOP Analysis is a well established method that identifies possible failure modes based on the interpretation of some standard keywords (No, More, Less, As well as, Part of, Reverse, Other than, Corrupt, Before/Early, After/Late). The possible failure modes in the different software functions or software function groups must be identified based on the systematic application of the keywords.

The keywords must be interpreted to derive failure modes. Following aspect has to be taken into consideration while interpreting the keywords:

- Required functionality (e.g. requirement is not / too much / too little / too late / … / fulfilled)
- Internal data structures or API parameters of a SW function (e.g. value is higher / lower / corrupted, or a data structure is smaller or bigger than expected)
- Program flow within a SW function (e.g. part of the program is executed)
- Timing of a SW function (e.g. function called too early, too late, before or after another function, on the same time as)

Then the interpreted keywords must be mapped to failure modes from the Safety Criticality Analysis.

11.6.4.2 Mapping of each failure mode to existing safety measure(s)
Identify effective safety measure(s) to prevent, detect and/or control the failure modes and their effects

If a new failure mode, which has not been listed in the regular Safety Criticality Analysis, is identified during the HAZOP Analysis, then this failure mode must be included in the Safety Criticality Analysis and a proper Safety Criticality Analysis must be performed for this failure mode. Likewise, if this leads to the point that the defined safety measures are insufficient to prevent, detect and/or control that failure mode, then new safety measures must be specified in the software technical safety concept.

11.7 SCA APPLIED TO MEDICAL USE CASE
In order to start a SCA, the prerequisites described above must be fulfilled. Therefore the software architecture and an allocation of the safety requirements to software components have to be defined. Hence, the use case describes in a first step its software architecture.
11.7.1 Description of the architecture

From former descriptions of the use case a more detailed chart of the data flow between the components was derived. It shows the basic components including some functions and shows a grid of connections used for the exchange of data.

![UML description of Medical use case](image)

Figure 20: UML description of Medical use case

-derived from the data flow chart was the architecture. The system is controlled by a core component. The other components are called from this core component. Shown in the graph are the safety levels.
11.7.1.1 Communication

The requirement
“System shall be able to send the stored data to authenticated person on request”
shall have a safety level C since a violation of this safety goal would cause the stored data to be
unavailable to an authenticated person or the authenticated person would get corrupted data. In case
of data unavailability the authenticated person could not make a diagnostic in urgent case. In case of
corrupted data, the authenticated person could make a wrong diagnostic.

Hazards can be:

- a temporarily unavailable transmission channel via LAN so no data can be transmitted on
  request
- a corruption of data because of failure in software or during LAN communication

Therefore, the components participating to the safety goal inherit the same safety level C:

- LAN Interface
- Authentication
- Encryption
- Firewall
- Data storage

The core component shall have level C as well since it controls the other components.

Safe State: Degraded operating mode:
11.7.1.2 Data collection and storage
The requirement

- System shall store sensor data into its internal memory shall have the safety level C because the violation of the safety goal would cause wrong data to be stored. This could lead later to a wrong diagnostic.

The hazard can be the corruption of any kind of medical data to be stored when needed.

Therefore, the components participating to the safety goal inherit the same safety level C:

- Data Collection
- Data integrity / validity
- Data Storage

If one of these components fails the medical data cannot be stored and is useless in consequence.

Safe State: If incoming data are not valid, generate a message for the patient.

11.7.1.3 System configuration
Once configured properly the configuration of the system is not critical to safe operation so a safety level of B is sufficient. In consequence an additional mechanism has to be applied in order to ensure that the system was properly configured i.e. a plausibility check.

Safe State: send an error message to the patient if a problem appears during the configuration process

11.7.1.4 Summary of the initial ASIL allocations
Based on the determination of the ASIL of the safety goals and their allocation to software modules of a preliminary architecture, it is now possible to define the corresponding safety measures required to ensure the criteria for coexistence (i.e. freedom from interference) and also try to decompose the requirements in order to tailor the ASIL.

11.7.2 Safety measures

11.7.2.1 Data validity
“Data Validity”: The “data Validity” shall protect the validated data with an index/timestamp, an ID (related to the type of data) and a checksum. Depending on the communication protocol, some of these safety mechanisms could be checked and removed of the payload before sending the data be-
cause already available in the protocol layer. But this would mean that the software implementing the protocol shall be in accordance to the safety level.

Consequence: the data flow from the “Data validity” up to the end user is protected against:

- Repetition
- Deletion
- Insertion
- Incorrect sequence
- Corruption
- Inconsistency

Masquerading is protected by the encryption.

Consequence: the modules Data storage, Data Integrity, Encryption, Firewall and LAN Interface Controller does not need to be protected against such kind of errors.

**Condition of Use:** The software receiving the data has to check the data integrity (checksum, index, ID)

11.7.2.2 Communication timing

Communication protocol:

- The receiver shall send an acknowledgment of the received messages
- Acknowledgment timeout detection shall be implemented on sender.
- A receiver timeout shall be implemented

Consequence: the protocol is protected against timing fault.

11.7.2.3 Control flow

As required for ASIL C software, a control flow has to be implemented. This will be done by using a watchdog (C)

**Consequence:** No external watchdog required (see Reference), only an internal one to support the control flow monitoring

11.7.2.4 Freedom from interference

As required by ISO26262, freedom from interference (spatial and temporal) shall be ensured between software modules with different ASIL.

Freedom from interference on data and hardware registers can be ensured with hardware memory protection.

Temporal freedom from interference is ensured using time independent tasks. Each task only contains software of one ASIL. The time independence of the tasks is ensured using HW timers (STM= system timer generating interrupts, temporal protection system (TPS) generating traps)

**Conclusion:** a scheduling system handling also the memory protection is required. This system is ASIL C
11.7.3 Requirement decomposition and ASIL tailoring

11.7.3.1 End-to-End protection

Implementation decision:

- the timeout detection is implemented in a software module independent from the “LAN Interface Controller” (this can be in the LAN Interface module or outside, but independent)
- the same software module also implement the check and generation of the safety measure of 1.

Let name this software module “End-to-End protection”

Then following decomposition is possible:

(C): System shall be able to send the stored data to authenticated person on request

(QM) System shall be able to send the stored data to authenticated person on request

(C) Any communication and storage errors shall be detected by the “End-to-End protection” and the system shall go to the safe state.

Thanks to this decomposition, following software modules can be developed as QM:

- LAN Interface
- Authentication
- Encryption
- Firewall
- Data storage
11.7.4 Safety Criticality analysis

The safety criticality analysis is performed on the authentication components in order to demonstrate how the method is used and how it can be generalized to the whole system.

11.7.4.1 Grouping of function

<table>
<thead>
<tr>
<th>Unit</th>
<th>Description</th>
<th>Parent</th>
<th>ASIL Target</th>
<th>Criticality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication module</td>
<td>Send a challenge with ID and challenge value to the client</td>
<td>Authentication module</td>
<td>ASIL-C</td>
<td>C3</td>
</tr>
<tr>
<td>Send challenge</td>
<td>return a new ID value not yet used for the CHAP</td>
<td>Send challenge</td>
<td>ASIL-C</td>
<td>C3</td>
</tr>
<tr>
<td>Auth_GetNewId</td>
<td>return a new challenge value not yet used for the CHAP</td>
<td>Send challenge</td>
<td>ASIL-C</td>
<td>C3</td>
</tr>
<tr>
<td>Auth_GetNewChallenge</td>
<td>Check the response of the client against the expected value and send the result passed/failed to the client. In case of fail, the communication session must be closed.</td>
<td>Authentication module</td>
<td>ASIL-C</td>
<td>C3</td>
</tr>
</tbody>
</table>
### Table 8: Medical - SCA functions grouping

#### 11.7.4.2 SCA

<table>
<thead>
<tr>
<th>Failure Modes</th>
<th>Possible Cause</th>
<th>Safety Measures</th>
<th>Description</th>
<th>ISO 26262 Reference</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>send wrong message length</td>
<td>variable of buffer length get corrupted or character indicating end of buffer is on the wrong place</td>
<td>check range of output parameters: buffer length</td>
<td>check that the information about the length of the transmitted data is correct</td>
<td>part 6, clause 7, table 4 1a</td>
<td>Range checks of input and output data</td>
</tr>
<tr>
<td>no challenge message sent</td>
<td>Corruption of the state of the authentication module state machine</td>
<td>double inverse storage of the state of the authentication module</td>
<td>If the computation of the hashing is not done in the same function, then a CRC should be added to the challenge string in order to detect corruption of the content.</td>
<td>part 6, clause 7, table 4 1c</td>
<td>Detection of data errors</td>
</tr>
<tr>
<td>no challenge message sent</td>
<td>message not accepted by the communication stack</td>
<td>check if message is accepted by communication stack</td>
<td>Check that the communication stack accepted to send the message.</td>
<td>part 6, clause 7, table 4 1e</td>
<td>Control flow monitoring</td>
</tr>
<tr>
<td>no challenge message sent</td>
<td>corruption of the stack or program counter</td>
<td>implement control flow on API level</td>
<td>For ASIL C, the control flow can be limited on API level.</td>
<td>part 6, clause 7, table 4 1c</td>
<td></td>
</tr>
<tr>
<td>wrong calculation of expected hash</td>
<td>corruption of challenge string before start of hashing</td>
<td>CRC on challenge string</td>
<td>If the computation of the hashing is not done in the same function, then a CRC should be added to the challenge string in order to detect corruption of the content.</td>
<td>part 6, clause 7, table 4 1c</td>
<td>Detection of data errors</td>
</tr>
<tr>
<td>wrong calculation of expected hash</td>
<td>corruption of secret before start of hashing</td>
<td>CRC on secret</td>
<td>The secret may be stored in Data flash and need to be requested by the authentication module. The CRC of the secret has to be checked before using the secret.</td>
<td>part 6, clause 7, table 4 1c</td>
<td>Detection of data errors</td>
</tr>
<tr>
<td>wrong calculation of expected hash</td>
<td>Error in hashing algorithm.</td>
<td>development process and testing</td>
<td>The hashing algorithm shall be checked with review and testing.</td>
<td>part 6, clause 7, table 4 1c</td>
<td>No plausibility or data range check is possible since the output is not known and can be inside the whole available data range. An ASIL D level could have required a diverse implementation or hardware.</td>
</tr>
<tr>
<td>wrong calculation of expected hash</td>
<td>calculation done with previous challenge string</td>
<td>check that the ID is different from the previous one</td>
<td>When a hashing is done, compare the ID with the one of the previous hashing. They must be different.</td>
<td>part 6, clause 7, table 4 1c</td>
<td></td>
</tr>
<tr>
<td>wrong comparison</td>
<td>comparison done with previous hashing result</td>
<td>check if the generated hash was already used in a comparison</td>
<td>Add a flag indicating that the hash string was already used for comparison. This flag can be reset when the new hash is generated.</td>
<td>part 6, clause 7, table 4 1c</td>
<td></td>
</tr>
</tbody>
</table>

**Project Partners:**
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
<table>
<thead>
<tr>
<th>Scenario</th>
<th>Reason</th>
<th>Action</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>wrong comparison</td>
<td>comparison done with previous response</td>
<td>check if the response was already used in a comparison</td>
<td>Add a flag indicating that the response string was already used for comparison. This flag can be reset when the new response is arriving.</td>
</tr>
<tr>
<td>wrong result of comparison</td>
<td>corruption of control flow inside the function</td>
<td>the result of comparison can only be available if the whole data has been compared</td>
<td>A result can only be given if the planned number of comparison steps has been done. Otherwise an error should be reported.</td>
</tr>
<tr>
<td>send wrong comparison result</td>
<td>No possible cause found</td>
<td>-</td>
<td>part 6, clause 7, table 4.1b Plausibility check</td>
</tr>
<tr>
<td>grant access to peer in case of failed comparison</td>
<td>No possible cause found</td>
<td>-</td>
<td>part 6, clause 7, table 4.1b Plausibility check</td>
</tr>
<tr>
<td>challenge message not sent in time</td>
<td>waiting for an information from an external module that is not responding</td>
<td>Implement timeout on all places where waiting trigger or information outside the authentication module is not available</td>
<td>It shall not happen that the software is in an infinite while loop waiting for another event to happen. An additional exit criteria related to a timeout shall be added.</td>
</tr>
<tr>
<td>challenge message not sent in time</td>
<td>corruption of the stack or program counter</td>
<td>implement control flow on API level</td>
<td>For ASIL C, the control flow can be limited on API level.</td>
</tr>
<tr>
<td>comparison not done in time</td>
<td>waiting for an information from an external module that is not responding</td>
<td>Implement timeout on all places where waiting trigger or information outside the authentication module is not available</td>
<td>It shall not happen that the software is in an infinite while loop waiting for another event to happen. An additional exit criteria related to a timeout shall be added.</td>
</tr>
<tr>
<td>comparison not done in time</td>
<td>corruption of the stack or program counter</td>
<td>implement control flow on API level</td>
<td>For ASIL C, the control flow can be limited on API level.</td>
</tr>
</tbody>
</table>

Table 9: Medical - SCA failure modes
11.8 SCA APPLIED TO AUTOMOTIVE E-MOTOR USE CASE

The SCA example is based e.g. on the ADC driver (part of MCAL layer). In order to start the SCA, the prerequisites described above must be fulfilled. Therefore the software architecture and an allocation of the safety requirements to software components have to be defined. Hence, the use case describes in a first step its software architecture.

11.8.1 Description of the architecture

The depicted architecture described the software component required to fulfill the e-motor functionality. It does not consider the safety requirements.

Torque control algorithm: ASIL D
11.8.3 Requirement decomposition for ASIL tailoring and safety measures

One of the ISO26262 requirements for ASIL D software is the usage of a diverse implementation. The purpose is to avoid systematic fault in the code that could lead to single point failures. So from this requirement, the consequence would be to implement diverse the ADC (analogue digital converter) driver.

Considering the fact that:

- The functional requirement of the ADC driver is to transfer a data from a hardware part to the upper layer
- The ADC is transferring two diverse data flow related to the same information

It comes out that there is another way to avoid, or at least detect, the systematic fault in the code.

Assuming that:

- The relation between the diverse data is not straightforward so that an error in the ADC driver cannot generate valid diverse data which are different from corresponding analog values
- The input signals are dynamic
- A validity checker is able to detect all types of possible systematic errors of the ADC driver, thanks to the properties of the dynamic and diverse input signals

It is therefore possible to state that the diversity in the input signals combined with a validity checker is enough to fulfil the ISO26262 requirement of diverse implementation for the ADC driver.

The requirement decomposition and ASIL tailoring can be expressed with following relation:

Read field information (ASIL D) = read signal 1 (ASIL B(D)) + read signal 2 (ASIL B(D)) + validity checker (ASIL D)

Consequences:

- The ADC driver functionality shall be developed according to ASIL B
- The two data flow (signal 1 and signal 2) with the help of the validity checker shall fulfil the ISO 26262 requirement of independence
- The validity checker functionality shall be developed according to ASIL D

The SCA shall confirm that the safety measures defined for the ADC driver are in accordance with the ISO26262 requirement for ASIL B software.

The DFA shall confirm that the safety measures defined on system level and in the ADC driver are adequate to fulfil the independence requirement of the two data flow.
11.8.4 Safety Criticality analysis

11.8.4.1 Grouping of function

The functionality of the ADC driver can be classified in following groups:

- ADC control (start / stop conversion)
- ADC read (read value, get status)
- ADC configuration (init, set buffer, enable / disable trigger)
- ADC support function (get version information)

11.8.4.2 SCA

<table>
<thead>
<tr>
<th>Failure Modes</th>
<th>Possible Cause</th>
<th>Safety Measures</th>
<th>Description</th>
<th>ISO 26262 Reference</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrupt Data</td>
<td>Data gets corrupted during the transfer from hardware register to buffer by:</td>
<td>Pointer Range Check</td>
<td>for memory corruption: the transfer of the data from hardware register to the software buffer is done in one C line and this is done on assembler level in two instructions. (DataBufferPtr = *CurrentResBufPtr.) The complexity of the line is very low; the corruption of the data during this transfer is very low. No additional mechanism required.</td>
<td>ISO 26262-6, Table 4: Detection of data errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Memory corruption by other SW</td>
<td></td>
<td>for unaligned and access of different result data formats</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Wrong interpretation and access of different result data formats</td>
<td></td>
<td>for unaligned memory access</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Unaligned memory access</td>
<td></td>
<td>Check the value of the pointer before reading the value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrupt Data</td>
<td>ADC result data is written to an unintended location (instead of into the result buffer), thereby corrupting memory space belonging to other software modules</td>
<td>Pointer Range Check</td>
<td>The range of the pointer access shall be determined separately and every pointer access shall be checked that it is within the allowed range. The check done inside ADC driver can only prevent against wrong computation or corruption inside the driver. We assume that the pointer value given by the upper layer is correct.</td>
<td>ISO 26262-6, Table 4: Range checks of input and output data</td>
<td>Runtime check is required, because this case cannot be detected reliably during code review, since the buffer size is defined by the application SW.</td>
</tr>
<tr>
<td>No Data</td>
<td>No result data in result buffer due to:</td>
<td>Data Available Check</td>
<td>A separate flag is maintained per group which indicates that the new data is available for reading. The GetGroupStatus API shall check this flag additionally before informing the user that the conversion status is COMPLETED</td>
<td>ISO 26262-6, Table 4: Diverse software design</td>
<td>Implement the check in Adc_GetGroupStatus</td>
</tr>
<tr>
<td></td>
<td>- No or aborted conversion</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Conversion &quot;hangs&quot;; therefore no data transfer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- SW did not transfer any data from result SFR to buffer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Data</td>
<td>No result data in result buffer due to:</td>
<td>Control Flow Monitor (CoU)</td>
<td>Implement control flow monitoring with verification of program sequence and watchdog timeout.</td>
<td>ISO 26262-6, Table 4: Control flow monitoring</td>
<td>Control Flow Monitoring (CoU) is expected to be implemented on application SW level</td>
</tr>
<tr>
<td></td>
<td>SW did not transfer any data from result SFR to buffer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, PTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, eesy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
| Old Data | Conversion Complete Check | Old data is transferred from the result SFR to the result buffer, i.e. the transfer was done before the conversion has been finished. | The ISR must check that the conversion is complete by verifying the VALID flag, before it transfers the data from the SFR to the result buffer. | ISO 26262-6, Table 4: Plausibility check |
| Old Data | Defensive Programming 1 | New conversion result data is available in the result SFR, but is not transferred to buffer (i.e less channels are transferred than converted) | Use only local variables in interrupt routine for ADC result transfer from result SFR to result buffer, to avoid transfer of old data. | Coding guideline |
| Old Data | Pointer Check | New conversion result data is available in the result SFR, but is not transferred to the buffer (i.e less channels are transferred than converted) | The ISR must check that the conversion is complete by verifying the VALID flag, before it transfers the channel results. | ISO 26262-6, Table 4: Plausibility check |
| Old Data | Data Available Check | Group Conversion status wrongly updated as "completed" thereby indicating to the application SW that conversion is complete. Therefore the user now gets old data when performing a read. | The GetGroupStatus API shall check a flag additionally before informing the user that the conversion status is COMPLETED | ISO 26262-6, Table 4: Diverse software design |
| Stuck Data | HW test (CoU) | Stuck-at fault in ADC HW. | Application must execute an ADC mux and conversion test to confirm proper HW function. | |
| Wrong Data | Pointer Redundant Storage | Reading/writing from/to wrong buffer address because of corruption of pointer | ResultBuffer address is stored redundantly. | ISO 26262-6, Table 4: Detection of data errors |
| Wrong Channel | Conversion Complete Check | Data from wrong channel is transferred to buffer and to application SW | The ISR must check that the conversion is complete by verifying the VALID flag, before it transfers the data from the SFR to the result buffer. | ISO 26262-6, Table 4: Plausibility check |
| Wrong Channel | Pointer Redundant Storage | Reading data from wrong buffer (that belongs to another channel group). | ResultBuffer address is stored redundantly | ISO 26262-6, Table 4: Detection of data errors |
| Unexpected Callback | Callback Plausibility Check (CoU) | Unexpected notification of application SW through callback function (e.g. due to group notification is not disabled). Possible timing impact. | The application SW should maintain the status of expected notification and ignore unexpected callbacks. | ISO 26262-6, Table 4: Plausibility check |
| Delayed Program Execution | Defensive Programming 2 | Unexpected delay or stall in any of the SW functions, due to: | No usage of endless loops (e.g. for polling). Use timeout checks or limited loops instead. | |
| Incorrect Sequence of Functions | Control Flow Monitor (CoU) | Application SW calls ADC functions in a wrong sequence or dependencies are not observed correctly (e.g. data, time, triggers, …) | Implement control flow monitoring with verification of program sequence and watchdog timeout. | ISO 26262-6, Table 4: Control flow monitoring |
| Corrupt Configuration | Configuration Check (CoU) | Corruption of ADC control registers (static configuration) | Regular verification of the static ADC configuration (ADC control registers). | |

Table 10: E-Motor SCA Failure modes
11.9 DESCRIPTION OF DEPENDANT FAILURE ANALYSIS (DFA)

The analysis method described below has to be performed during the software architecture phase (ISO26262 part 6 clause 7). The aim of this analysis is to identify single cause that can invalidate a required freedom from interference between two software elements and lead to the violation of a safety requirement or safety goal.

The analysis has to be performed according to the description in part 9, clause 7. The Dependent Failure Analysis is a qualitative analysis method to verify the Software Architecture and SW Safety Concept.

The aim of this analysis is to identify single cause that can invalidate a required freedom from interference or independence between two software elements and lead to the violation of a safety requirement or safety goal.

This type of analysis is done on software elements that could be affected by common cause failure or cascading failure. Based on a list of root cause and common cause initiator, each module is analyzed.

11.9.1 Types of dependent failure

Dependent failures are defined by ISO 26262 as failures whose probability of simultaneous or successive occurrence cannot be expressed as the simple product of the unconditional probabilities of each of them.

ISO26262 distinguish two types of dependent failures.

11.9.1.1 Common Cause Failure

![Diagram of Common Cause Failure]

Figure 24: Description of Common Cause Failure
A common cause initiator exists and triggers the same or different systematic software faults (2 and 2') in both SW elements, causing both to fail. The combination of the resulting failures leads to the violation of the safety goals.

11.9.1.2 Cascading Failure

![Diagram of cascading failure]

Figure 25: Description of Cascading Failure

A fault (1) in SW element 1 leads to its failure. Due to insufficient fault containment in SW element 1 or insufficient independence of SW element 2 a coupling mechanism (2) exists, which leads to a failure of SW element 2. The combination of the two resulting (cascading) failures or the resulting (cascading) failure 2 leads to a violation of the safety goal (3).

11.9.2 DFA Prerequisites

To perform a dependent failure analysis on module level, following documents need at least to be available:

- Software safety requirements (resulting from ISO26262 part6, 6.5.1)
- Software architectural specification (resulting from ISO26262 part6, 7.5.1)

Please see “11.6.1 SCA Prerequisites” for more information.
11.9.3 Identification of software elements or functions for DFA

These software elements that can be affected by common cause initiator are typically:

- The redundant or diverse elements (e.g. resulting from an ASIL decomposition)
- The software module and its safety mechanisms (e.g. software functional part and the watchdog)
- The software functions using identical SW modules (e.g. libraries)
- The software that rely on the same hardware or same input signal

11.9.4 Identification of Dependent Failures

ISO 26262 is giving a list of possible root causes for hardware and software dependent failures. The list can be reduced to the ones related to software:

- Random hardware failures (e.g. CPU and memory structures)
- Development faults (e.g. during software development process)
- Installation faults (e.g. during configuration, integration)
- Environmental factors (e.g. impact on input values, interrupts)
- Failures of common internal and external resources (e.g. libraries)

Based on the list, the possible CCI and software fault must be retrieved.

On order to make a more systematic analysis about the possible CCI or software fault, we first propose to list the type of software properties that can be affected by the root causes:

Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adeland, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Electricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
Based on the matrix of root causes and affected software properties, a better systematic analysis of the possible CCI, software fault or the resulting failures can be done.

### 11.9.4.1 Identification of Common Cause Failure

During this analysis phase, it is required to think about the possible CCI or about the resulting failures that could be related to the root cause and that affect the listed software properties. Some examples are given in the table below.

<table>
<thead>
<tr>
<th>RootCause</th>
<th>SW properties</th>
<th>Random-HW failures</th>
<th>Development faults</th>
<th>Installation faults</th>
<th>Environment faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>shared data</td>
<td>unexpected signals combination</td>
<td>wrong data configured</td>
<td>not expected input values</td>
<td></td>
</tr>
<tr>
<td>Timing</td>
<td>too many interrupts</td>
<td>unexpected signal period</td>
<td>NA</td>
<td>too many interrupts</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 27: Matrix for Common Cause Failure**

For each possible violation of a safety goal, find out what are the failures and what is the CCI, which trigger the common cause faults to cause a common cause failure.

### 11.9.4.2 Identification of Cascading Failure

During this phase, it is required to think about the possible fault in a software module and the possible coupling mechanisms that could lead to a cascading failure.

In the case of cascading failure, the failure can only originate from a software module, i.e. the root cause is a single point failure in software. Therefore the analysis is reduced to the ISO26262 root cause “development faults”. The table below gives some examples about possible faults or failures to be considered for the cascading failure analysis.
11.9.5 Specification of Measures against Dependent Failures

For each set of

- CCI, faults, failures, violation of safety goal
- Fault, coupling mechanism, failure, violation of safety goal

It is required to check in the safety concept the existence of or to define a measure to:

- Prevent CCI or root cause
- Avoid or reduce the systematic fault
- Improve the fault containment
- Control the failure

Remark:
The verification of the effectiveness of the specified measures is done qualitatively, e.g. via reference to the appropriate methods recommended for the respective ASIL requirements.

11.10 DFA APPLIED TO MEDICAL USE CASE

The DFA is applied, as for the SCA, on the authentication module.
The first step is to define the elements that have to be considered for the dependent failure analysis:

Figure 28: Matrix for Cascading Failure
• due to decomposition QM + ASIL C: Communication stack QM
• safety mechanism: watchdog
• possible common libraries: encryption algorithm
• common HW: encryption HW accelerator
• software with different ASIL: LAN interface, Firewall, Encryption, data storage are QM

11.10.1 Common Cause Failure Analysis

<table>
<thead>
<tr>
<th>Root Cause</th>
<th>SW Domain</th>
<th>CC Initiator / CC Weakness</th>
<th>Failure Mode</th>
<th>Impact on Application</th>
<th>Safety Measure</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Hardware</td>
<td>Data</td>
<td>HW failure is modifying the content of the RAM</td>
<td>Malfunction of the authentication module and of the watchdog or other safety mechanisms required by the authentication module</td>
<td>The safety mechanisms cannot detect that the authentication module has a failure. No possibility to go to the safe state</td>
<td>HW: add correcting code on RAM and flash content. (See HW safety mechanism ECC)</td>
<td>Fault detection/control</td>
</tr>
<tr>
<td>Failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random Hardware</td>
<td>Data</td>
<td>HW failure is corrupting the execution of an CPU operation</td>
<td>Malfunction of the authentication module and of the watchdog or other safety mechanisms required by the authentication module</td>
<td>The safety mechanisms cannot detect that the authentication module has a failure. No possibility to go to the safe state</td>
<td>HW: use a lockstep CPU</td>
<td>Fault detection/control</td>
</tr>
<tr>
<td>Failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Development Fault</td>
<td>Data</td>
<td>SW failure outside authentication module is modifying the content of the RAM</td>
<td>Malfunction of the authentication module and of the watchdog or other safety mechanisms required by the authentication module</td>
<td>The safety mechanisms cannot detect that the authentication module has a failure. No possibility to go to the safe state</td>
<td>SW: use Memory Protection Unit (MPU) or double inverse storage to protect the safety mechanisms and the authentication module from spatial SW interference of modules with lower ASIL. Moreover, if MPU is used, the variables from authentication module and the corresponding safety mechanism shall not be in the same partition.</td>
<td>Fault containment</td>
</tr>
<tr>
<td>Development Fault</td>
<td>Control Flow</td>
<td>SW failure outside authentication module is modifying the content of the stack</td>
<td>Malfunction of the authentication module and of the watchdog or other safety mechanisms required by the authentication module</td>
<td>The safety mechanisms cannot detect that the authentication module has a failure. No possibility to go to the safe state</td>
<td>SW: the authentication module and the corresponding safety mechanism shall not be executed using the same stack and heap.</td>
<td>Fault containment</td>
</tr>
<tr>
<td>Development Fault</td>
<td>Control Flow</td>
<td>processing of data based on interrupt</td>
<td>in case too many interrupt take place, it is not possible to handle them all</td>
<td>The peer requesting an authentication does not get a response</td>
<td>Base the computation on cyclic triggered task</td>
<td>Fault avoidance</td>
</tr>
</tbody>
</table>

Table 11: Medical - Common cause failure

11.10.2 Cascading Failure Analysis

<table>
<thead>
<tr>
<th>Root Cause</th>
<th>SW Domain</th>
<th>Coupling Mechanism</th>
<th>Failure Mode</th>
<th>Impact on Application</th>
<th>Safety Measure</th>
<th>Category</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Fault</td>
<td>Data</td>
<td>RAM</td>
<td>Another piece of software is corrupting the RAM variables of Authentication module</td>
<td>Could violate safety goal</td>
<td>protect RAM variable with double inverse storage or memory protection</td>
<td>Fault containment</td>
<td></td>
</tr>
</tbody>
</table>
### Development Fault

<table>
<thead>
<tr>
<th>Type</th>
<th>Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>common HW / external resources</td>
<td>ADC common registers</td>
</tr>
<tr>
<td>common HW / external resources</td>
<td>Common tool to generate the ADC configuration</td>
</tr>
<tr>
<td>decomposition</td>
<td>two independent ASIL B(D) ADC data paths (within the same ADC module)</td>
</tr>
<tr>
<td>decomposition</td>
<td>ADC plausibility check in upper layer</td>
</tr>
<tr>
<td>decomposition</td>
<td>registers of independent ADC clusters</td>
</tr>
<tr>
<td>identical SW</td>
<td>common configuration parameter of ADC module</td>
</tr>
<tr>
<td>identical SW</td>
<td>Common library: disable/enable interrupt, ...</td>
</tr>
<tr>
<td>identical SW</td>
<td>Common initialization of ADC</td>
</tr>
<tr>
<td>partition</td>
<td>software modules with different ASIL</td>
</tr>
<tr>
<td>safety mechanisms</td>
<td>Safety mechanism e.g watchdog, MPU configuration</td>
</tr>
</tbody>
</table>

Table 12: Medical - Cascading Failure

### 11.11 DFA Applied to Automotive E-Motor Use Case

The DFA is applied, as for the SCA, on the ADC module.
The first step is to define the elements that have to be considered for the dependent failure analysis:
### 11.11.1 Common cause failure analysis

<table>
<thead>
<tr>
<th>Root Cause</th>
<th>SW Domain</th>
<th>CC Initiator / CC Weakness</th>
<th>Failure Mode</th>
<th>Impact on Application</th>
<th>Safety Measure</th>
<th>Category</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Hardware Failure</td>
<td>Shared HW Resources</td>
<td>common register for the independent ADC clusters</td>
<td>An unexpected modification of these registers due to e.g. HW soft error</td>
<td>Application does not get any result or delayed or a wrong result.</td>
<td>CoU: the plausibility check shall detect such errors. Mechanism: cyclic check of common registers against predefined fix value.</td>
<td>Fault detection/control</td>
<td>A cyclic check is easy to implement if the common register values are fix after initialization. Actually a simple reading of the data is enough to check corruption from HW error since the ECC is check as soon as the data is read.</td>
</tr>
<tr>
<td>Random Hardware Failure</td>
<td>Code &amp; Configuration</td>
<td>Common code and configuration for ADC driver</td>
<td>An corruption of the content due to HW error</td>
<td>ADC driver gives no or wrong data to upper layer. ADC driver corrupt other ADC channels</td>
<td>CoU HW: usage of Error Correcting code for flash</td>
<td>Fault detection/control</td>
<td>Is not time consuming for SW</td>
</tr>
<tr>
<td>Random Hardware Failure</td>
<td>Data</td>
<td>common variables for the independent ADC clusters</td>
<td>An unexpected modification of these variables (HW soft error)</td>
<td>Application does not get any result or delayed or a wrong result.</td>
<td>CoU HW: usage of Error Correcting code for RAM</td>
<td>Fault detection/control</td>
<td>Is not time consuming for SW</td>
</tr>
<tr>
<td>Random Hardware Failure</td>
<td>Shared HW Resources</td>
<td>same HW trigger for ADC conversion</td>
<td>The HW trigger does not take place, or not at the right time</td>
<td>Application does not get the result or at the wrong time</td>
<td>CoU: the plausibility check shall detect such errors. CoU: use different HW trigger for the independent ADC paths. The plausibility check can then detect the difference in the data</td>
<td>Fault detection/control</td>
<td></td>
</tr>
<tr>
<td>Development Fault</td>
<td>Data</td>
<td>no independence of data between the ASIL B(D)</td>
<td>Data corrupted by another software module. Both data read from ADC are corrupted or the ADC functionality fails</td>
<td>The application is getting wrong or no data from ADC driver</td>
<td>CoU: the plausibility check shall detect such errors. Since data are diverse, a corruption of data can be detected. The plausibility check shall also check violation of the time requirements. Mechanism: variables from ADC driver belonging to the 2 independent ASIL B(D) paths shall be stored double inverse</td>
<td>Fault detection/control</td>
<td>Double inverse storage is not efficient in this case because if the ADC driver is accessing the wrong ADC channel, the double inverse storage will also be correctly written by the driver.</td>
</tr>
<tr>
<td>Development Fault</td>
<td>Data</td>
<td>no independence of data between the ASIL B(D)</td>
<td>Data corrupted by the ADC driver. Both data read from ADC are corrupted or the ADC functionality fails</td>
<td>The application is getting wrong or no data from ADC driver</td>
<td>CoU: the plausibility check shall detect such errors. Since data are diverse, a corruption of data can be detected. The plausibility check shall also check violation of the time requirements. Mechanism: CoU: variables from ADC driver belonging to the 2 independent ASIL B(D) paths shall be protected with memory protection.</td>
<td>Fault detection/control</td>
<td></td>
</tr>
</tbody>
</table>
## D3.1 – Specification of Safety and Security Analysis and Assessment Techniques

### Table 14: Automotive E-Motor - Common Cause Failure

| Devel- | Shared HW| common | An unexpected | The application | CoU: the plausibility check shall detect such errors. | Fault detection/contr. |
|opment | Resources| register| modification of these registers due to SW failure| is getting wrong (includes at wrong time) or no data from ADC driver| Mechanism: cyclic check of common registers against predefined fix value. | A cyclic check is easy to implement if the common register values are fix after initialization. In this case, the data shall be checked against a predefined value since the corruption is done per software, hence checking the ECC is not enough. |
| Fault | ADC clusters | | | | | |
| Devel- | registers of the ADC clusters are not protected against interference | An unexpected modification of these registers due to SW failure | The application is getting wrong | CoU: the plausibility check shall detect such errors. | CoU: the registers of the independent ADC paths shall be protected with Memory protection | Fault detection/contr. |
|opment | Code & Configuration | the same code is used for the independent ADC paths during runtime | the same error in the control flow will affect both signal with the same failure | The plausibility check may not detect this control flow error | No safety mechanism is required because: the data to transfer are diverse | Fault detection/contr. |
| Fault | the same code is used for the independent ADC paths during initialisation | the same error in the control flow will affect both signal with the same failure | The plausibility check may not detect this control flow error | mechanism: implement an independent code that verifies the initialization based on the configuration data stored in flash | | |
| Devel- | Code & Configuration | the same code is used for the independent ADC paths during runtime | the same error in the control flow will affect both signal with the same failure | The plausibility check may not detect this control flow error | | |
|opment | Failure | the same configuration set is used to configure both independent ADC paths | the configuration is wrong | both independent channels are not correctly initialized | The integrator and configurator of the project shall check the ADC configuration | Fault avoidance |
| of Common | Code & Configuration | the same configuration set is used to configure both independent ADC paths | the configuration is wrong | neither independent channels are not correctly initialized | | |
| Resources | | | | | | |

### 11.11.2 Cascading Failure Analysis

<table>
<thead>
<tr>
<th>Root Cause</th>
<th>SW Domain</th>
<th>Coupling Mechanism</th>
<th>Failure Mode</th>
<th>Impact on Application</th>
<th>Safety Measure</th>
<th>Category</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop-</td>
<td>Data</td>
<td>no independence of ADC driver internal variables between the ASIL B(D) data paths</td>
<td>Data corrupted by the ADC driver, both data read from ADC are corrupted or the ADC functionality fails</td>
<td>The application is getting wrong or no data from ADC driver</td>
<td>CoU: the plausibility check shall detect such errors.</td>
<td>Fault detection/contr.</td>
<td>If the ADC driver fault is in the computation of the ADC channel, then a double inverse storage of internal ADC variables will not help. But it is a very low probability that both independent channels are corrupted at the same time. So the plausibility checker should detect such corruption</td>
</tr>
<tr>
<td>ment Fault</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Project Partners:
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
### Development Fault

**Data**

- no independence of ADC driver internal fix variables between the ASIL B(D) data paths
- Fix data corrupted by or outside the ADC driver.
  - The application is getting wrong or no data from ADC driver
  - CoU: the fix data (fix after initialization) shall be protected by memory protection

| Fault avoidance | This can be easily done and does not cost an additional MPU region if the variables are correctly ordered in the memory space |

| Development Fault | Time | no time independence between both ASIL B(D) | a time violation in one path would interfere with the second path | The application is getting no data from ADC driver | CoU: the ADC values of the both data paths shall be retrieved in time independent requests | Fault containment |

### Development Fault

**Control Flow**

- no independence in the stack and heap
- a fault in the control flow of the first data path can interfere with the stack and heap of the second independent control flow
  - The application is getting wrong or no data from ADC driver
  - CoU: the ADC control flow of both data path shall not be able to interfere with the stack of the other one. This can be achieved by time separation or memory protection

### Development Fault

**Shared HW Resources**

- the registers of the independent ADC clusters can be accessed by the control flow of both data paths
  - On control flow can corrupt registers of both independent ADC clusters
  - The application is getting wrong or no data from ADC driver
  - CoU: the registers of the ADC clusters shall be protected by memory protection

Table 15: Automotive E-Motor - Cascading Failure

### 11.12 Possible relation with security – Outlook

The safety analysis performed on the medical and automotive e-motor use cases did only consider unintended fault of hardware or software. Hence no claim can be made about how good the safety measures defined could also contribute to the safety properties of the system.

A possible extension of these safety analyses could be to check if additional failure modes, common cause initiator or weaknesses and coupling mechanisms should be taken into account when considering also security attacks.

A second type of extension would be to check the influence of the defined safety mechanisms on security. The usage of a Markov chain may help to consider and point out the synergies or trade-offs between safety and security properties.

Finally, the assessment aspects also need to be taken into consideration. The safety analysis methods described above are some of the possible analyses required to comply with ISO26262. Up to now, no security standard is defined for the automotive domain. Could an extension to security of the described analysis methods be part of a future safety-security standard?

Could the ISO/IEC 15408 approach, also known as "Common Criteria", be a framework for security analysis in automotive?

### 11.13 Reference

“Common Cause Analysis for Integrated Circuits with On-Chip Redundancy”, Rainer Faller, Safetronic 2010
11.14 **VOCABULARY**

<table>
<thead>
<tr>
<th>Safe State</th>
<th>ISO26262: operating mode of a system without an unreasonable level of risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>QM</td>
<td>Quality Management = no safety related development</td>
</tr>
<tr>
<td>Independence</td>
<td>absence of dependent failure (this has to be confirmed by a Dependent Failure Analysis)</td>
</tr>
</tbody>
</table>

11.15 **ANNEX A. COMMUNICATION FAULT MODEL**

Following fault model can be considered for a communication network (EN 50159-2 table C.2)

- Repetition
- Deletion
- Insertion
- Incorrect sequence
- Corruption
- Timing fault
- Inconsistency
- Masquerading

11.16 **ANNEX B. MECHANISMS FOR ERROR DETECTION (REF. ISO 26262)**

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL A</th>
<th>ASIL B</th>
<th>ASIL C</th>
<th>ASIL D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Range check of input data</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1b Plausibility check</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>1c Detection of data errors</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>1d External monitoring facility</td>
<td>o</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>1e Control flow monitoring</td>
<td>o</td>
<td>+</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1f Diverse software design</td>
<td>o</td>
<td>o</td>
<td>+</td>
<td>++</td>
</tr>
</tbody>
</table>

Table 16: Mechanisms for error detection at the software architectural level (ref: ISO 26262)

---

**Project Partners:**
Intecs, Institute of Informatics and Telematics - CNR, AKHELA, Università degli Studi di Roma La Sapienza, Technical University of Denmark, FTW Forschungszentrum Telekommunikation Wien, Adelard, UniControls, Czech Technical University in Prague, PSA Peugeot Citroën, SYSGO, ikv++ Technologies, easy-id, Infineon Technologies AG Deutschland, EADS DEUTSCHLAND, Électricité de France, SYSGO s.r.o., Siemens AG Österreich, City University London - Centre for Software Reliability, General Motors Research & Development

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Partners accept no liability for any error or omission in the same.
12 CONCLUSIONS

This document has provided a specification of safety and security analysis and assessment techniques as detailed in the technical annex:

“Specification of Safety and Security Analysis and Assessment Techniques: The preliminary version D3.1 will focus on the definition of the safety and security metrics, and on the definition of the methodologies and techniques for the assessment of the safety and security properties.”

We believe that this document provides the necessary basis for the consortium to select, develop and apply those methods and techniques that seem most promising for dealing with the SESAMO use cases as identified in deliverable D1.1.

The selection of the methods and techniques reported upon were based on the competences of the partners, focusing on the more novel or well-developed directions available within the consortium, and dispensing with a treatment of methods and techniques that are standard within the area and not particularly addressing the primary challenge of SESAMO: the need to consider safety and security at the same time.

To facilitate the adoption of the methods and techniques presented, each section gave an overview of a particular approach. It then made a feasibility study of applying the methods and techniques to the SESAMO use cases and how to achieve some of the concrete aims of the use cases.

In the next period a more substantial testing of the ideas will be made and the up-coming deliverable D3.2 will reflect the choices of methods and techniques that seems most fruitful for reaching the objectives of SESAMO.