
# IoT4CPS – Trustworthy IoT for CPS

FFG - ICT of the Future
Project No. 863129

# Deliverable D3.7
# Final Report on Design & Methods

The IoT4CPS Consortium:

AIT – Austrian Institute of Technology GmbH
AVL – AVL List GmbH
DUK – Donau-Universität Krems
IFAT – Infineon Technologies Austria AG
JKU – JK Universität Linz / Institute for Pervasive Computing
JR – Joanneum Research Forschungsgesellschaft mbH
NOKIA – Nokia Solutions and Networks Österreich GmbH
NXP – NXP Semiconductors Austria GmbH
SBA – SBA Research GmbH
SRFG – Salzburg Research Forschungsgesellschaft
SCCH – Software Competence Center Hagenberg GmbH
SAGÖ – Siemens AG Österreich
TTTech – Auto AG
TTTech – TTTech Computertechnik AG
IAIK – TU Graz / Institute for Applied Information Processing and Communications
ITI – TU Graz / Institute for Technical Informatics
TUW – TU Wien / Institute of Computer Engineering
XNET – X-Net Services GmbH



*For more information on this document or the IoT4CPS project, please contact:*
Mario Drobics, AIT Austrian Institute of Technology, mario.drobics@ait.ac.at

## Document Control

**Title:** Design & Methods Concept
**Type:** Public
**Editor(s):** Stefan Jaksic
**E-mail:** Stefan.Jaksic@ait.ac.at
**Author(s):** Abdelkader Shabaan, Wolfgang Herzner, Stefan Jaksic (AIT), Martin Matschnig, Lukas Krammer, Daniel Hauer (Siemens), Leo Botler (TUG ITI),
**Doc ID:** D3.7

## Amendment History

| Version | Date | Author | Description/Comments |
|---|---|---|---|
| V0.1 | 19.05.2020 | S. Jaksic | Initial version prepared |
| V0.2 | 17.06.2020 | S. Jaksic, L. Botler, L. Krammer, D. Hauer | Recommender System, SHSA, Trustworthy localization |
| V0.3 | 19.06.2020 | S. Jaksic, W. Herzner | Appendix pattern for SHSA |
| V0.4 | 22.06.2020 | S. Jaksic, W. Herzner | Review comments |
| V1.0 | 23.06.2020 | S. Jaksic | Consolidating the text |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |



The IoT4CPS project is partially funded by the "ICT of the Future" Program of the FFG and the BMK.

Federal Ministry
Republic of Austria
Climate Action, Environment,
Energy, Mobility,
Innovation and Technology

FFG
Forschung wirkt.

# 1. Contents

## List of Abbreviations

| | |
|---|---|
| AoA | Angle of Arrival |
| API | Application Programming Interface |
| AT | Attack Tree |
| ATA | Attack Tree Analysis |
| BAS | Basic attack steps |
| BLE | Bluetooth Low Energy |
| CAM | Context Aware Monitoring |
| COTS | Commercial of the Shelf |
| CR | Component Requirement |
| CPS | Cyber-Physical System |
| CPU | Central Processing Unit |
| DFD | Data Flow Diagram |
| DI | Data Input |
| DO | Data Output |
| DoA | Direction of Arrival |
| DoS | Denial of Service |
| DNF | Disjunctive Normal Form |
| EA | Enterprise Architect |
| EDR | Embedded Device Requirements |
| FMEA | Failure Mode and Effects Analysis |
| FMVEA | Failure Mode, Vulnerabilities and Effects Analysis |
| FPGA | Field Programmable Gate Array |
| FTA | Fault Tree Analysis |
| HAS | Higher attack states |
| I4.0 | Industry 4.0 |
| IoT | Internet of Things |
| IIoT | Industrial IoT |
| KP | Knowledge Pack |
| MORETO | Model-based Security Requirement Management Tool |
| NDR | Network Device Requirements |
| OSF | Open Semantic Framework |
| PDoA | Phase Difference of Arrival |
| QoS | Quality of Service |
| ROTS | Real-Time Operating System |
| RTLS | Real-Time Localization Systems |
| SA | Security Achieved |
| SCPN | Stochastic Colored Petri Net |
| ST | Security Target |
| SHSA | Self-Healing through Structural Adaptation |
| SysML | System Modelling Language |
| TARA | Threat Analysis and Risk Assessment |
| TOE | Target of Evaluation |
| TS | Target System |
| UC | Use Case |
| UML | Unified Modelling Language |
| UWB | Ultra Wide Band |
| VBA | Visual Basic |

| | |
|---|---|
| V&V | Verification and Validation |
| WCET | Worst Case Execution Time |
| WP | Work Package |
| XML | eXtensible Markup Language |

Executive Summary

This deliverable summarizes the innovative methods and tools for achieving dependable Internet of Things (IoT) systems, which were developed by scientific and industrial partners of IoT4CPS project. IoT4CPS project provides guidelines, processes and recommendations to build dependable IoT systems. Presented methods and tools tackle challenges along different Cyber-Physical System (CPS) architecture layers: information layer, control layer and network layer. In this deliverable we also position WP3 contributions w.r.t. IoT4CPS use cases: Automated Driving and Industry4.0.

The physical-level tools and methods such as sensor security measures for discovering faulty and hacked sensors are reported in deliverable D3.4: “System architecture patterns for enabling multi-stakeholder trust provisioning during production and maintenance”. In addition, we report on forward-secure key exchange mechanism in our deliverable D3.6.2: “Prototype cryptographic library implementation”. On a network level we report on recommender systems to develop dependable IoT system, which can help users who want to build large IoT systems to choose the appropriate protocols and system configurations. Another method to achieve dependability, applied on a platform-level, is the Self-Healing by Structural Adaptation which allows systems to leverage implicit redundancy to achieve resiliency to failures. We also report on novel solutions for trusted localization and orientation and provide a fair cross-technology comparison. Based on our thorough practical experience during designing and implementing the experimental setup we provide recommendations and guidelines to be used by researchers and developers in this domain.

On an application level we report on tools for a variety of tasks in cyber security. We present ThreatGet, a tool that identifies, detects, and understands potential security threats in the foundation level of system models. Moreto is a tool for security requirements analysis and management using modelling languages such as SysML/UML. Our next contribution is a tool for standard-based product development management: GSFlow. It is one of the results of a more general effort to develop tools to support model-based development approaches and Safety & Security by Design. Last but not the least, we report on methods for safety and security risk assessment and formalize it into a verification pattern.

## 2. Introduction

Deliverable D3.7: Final Report on Design & Methods the last sequel of our series of deliverables in work package 3 of IoT4CPS project, has Deliverable *D3.2: Guidelines, processes and recommendations for the design of dependable IoT Systems* as a direct predecessor. This deliverable is mostly focusing on research conducted as a part of task *T3.1: Dependability design methods for IoT*. In D3.7 we further elaborate methods from D3.2 and explain how a user can leverage those methods to obtain dependable systems. We also upgrade the deliverable with new methods and tools such as Resilience analysis of indoor direction-finding technologies, Verification and Validation (V&V) patterns for Self-Healing by Structural Adaptation (SHSA). We also explain how one can extend SHSA method by Context Aware Monitoring (CAM).

In order to classify different kinds of methods and tools, it is important at this point to provide the reference to the architectural layers of the V-model. Similarly to D3.2, in this deliverable we also collect and present potential solutions, tools and methodological building blocks for the development of safe and secure Internet of Things (IoT) and Cyber-Physical Systems (CPS). This deliverable puts a special focus on the integration of privacy and on the support of the complete engineering cycle, from engineering support to providing potential solutions.

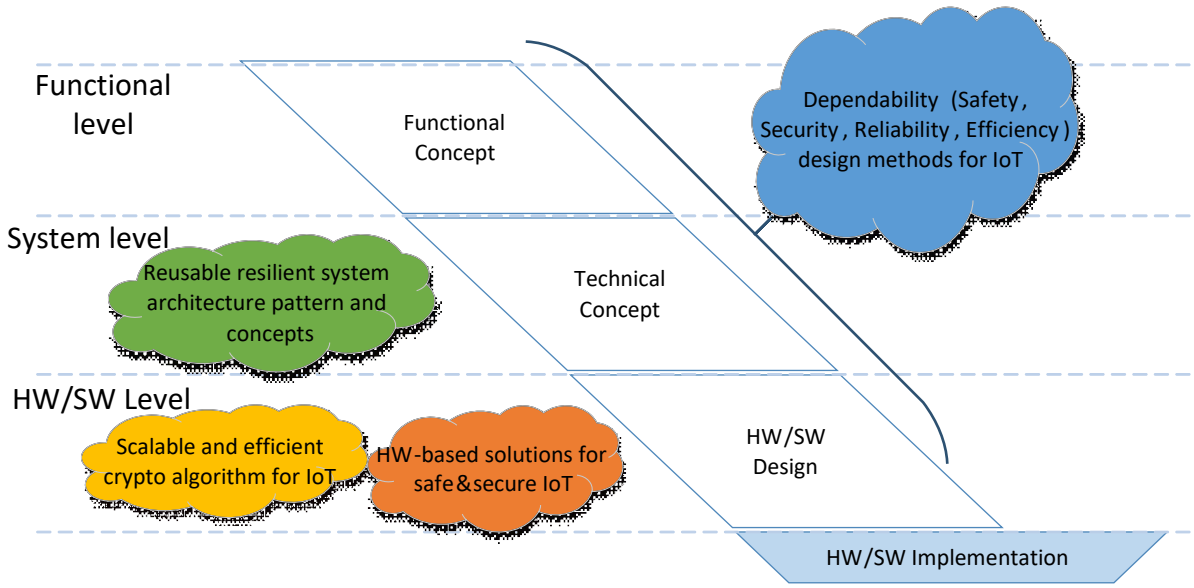


**Figure 1: Structuring WP3 contributions along the V-Model**

The contributions are based around the left side of the V-Model. The usage of the V-Model is here mainly as a structuring model and for easier classification and explanation. The usage of the V-Model should not be understood as an enforcement of this process model and the presented methods and building blocks are not restricted in term of engineering process model.

Our understanding of dependability itself is based on [39]. In this work, the dependability is differentiated between attributes, threats and means. The attributes summarize all parts of dependability, e.g. the set of properties we would like to ensure that a system we need to rely on possess. Due to the need of our industrial partners as well as the nature of our two main use cases, the IoT4CPS project sets priority on safety and security.

Threats are potential factors which can cause a violation or contribute to a violation of a dependability attribute. In order to protect the attributes, we can use different means. The following section will give a short overview about the basic system concept considered in IoT4CPS and present then different means to achieve dependability, which are considered in WP3 of IoT4CPS project.

Our tools, methods and guidelines are mostly non-directly bound to a specific use case and can be applied to either of our IoT4CPS main use cases: Industry 4.0 as well as Autonomous Driving. Two of them which are tightly coupled to the use case are the Autonomous Driving Platform developed by TTTech, as well as the Recommender System for IoT, developed by Siemens, for our Industry 4.0 use-case. The overview can be seen in Figure 2.

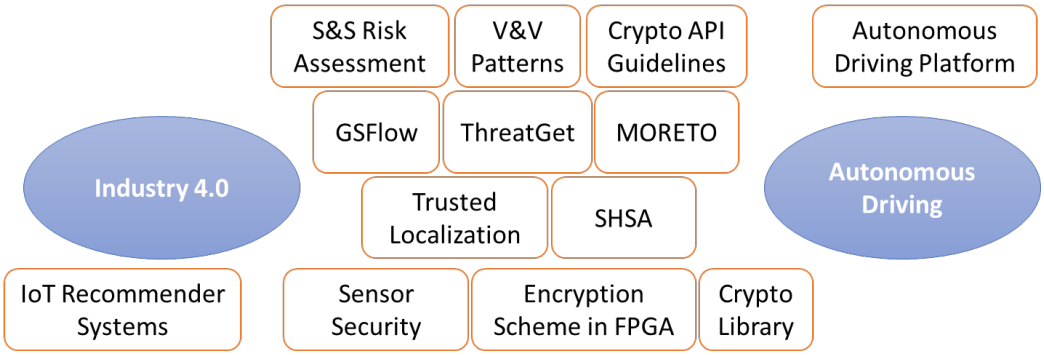


**Figure 2: Most contributions of WP3 fits into both of IoT4CPS use-cases**

## 3. Overview and System Model

In Figure 3 we can see how different WP3 contributions are positioned in the CPS Layer stack. A typical CPS is a complex system of systems which interact with one another. Depending on their purpose the components are usually classified into one of four layers: information layer, control layer, network layer and physical layer. The physical level is the lowest level in hierarchy. This is where data is obtained from sensors and data is prepared to be sent to other systems (i.e. encrypted). Once the system obtained the data, it is accessed and transmitted to interested components on network nodes. Typically, this takes place on network layer. On top of network layer, we see the platform layer. It is a layer where most system core functions are implemented, which are to be used by application layer.

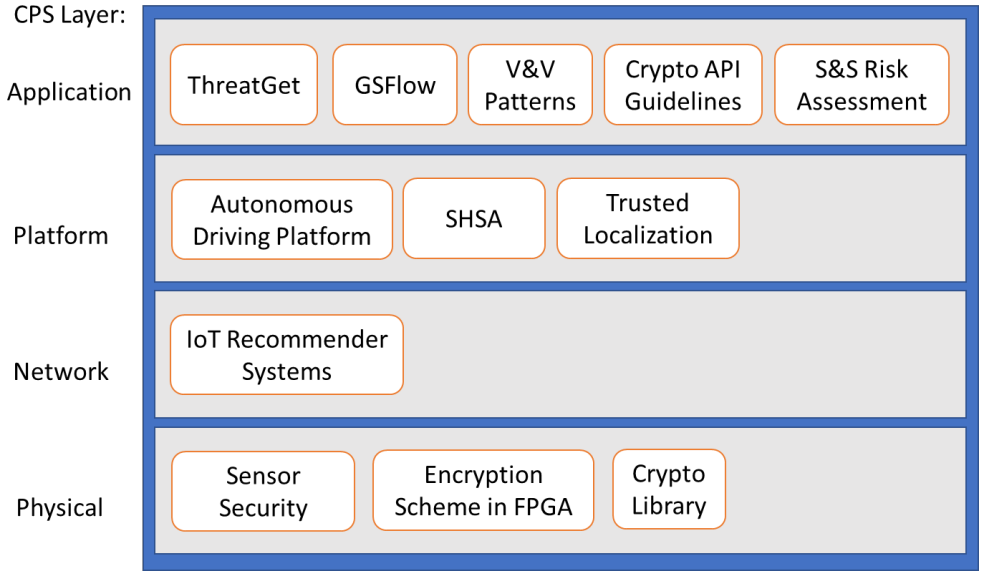


**Figure 3: Contributions to safety and security can be separated into different CPS layers.**

IoT4CPS project achievements can be found in any of the CPS architecture layers. On application layer, we have our V&V patterns and cyber security tools such as GSFlow, Moreto and ThreatGet. In addition, the guidelines for developing usable cryptographic APIs by SBA research belongs to the CPS application layer. Finally, we report on hybrid methods for safety and security risk assessment.

Underneath the application layer, in the platform layer we find Self-Healing for Structural Adaptation (SHSA) a technique for building resilient CPS which is developed by TU Wien. Solutions for trusted localization and orientation in space, useful in our Industry 4.0 use case are developed by TU Graz together with JKU Linz. Finally, we have an autonomous driving platform which is developed by TTTech. To build a reliable and functional IoT ecosystem one can use a recommender system, developed by Siemens. Last but not the least, AIT and TUG developed low-overhead encryption scheme, which is also implemented in FPGA by Siemens. DUK will provide concepts for achieving sensor security and low-level data integrity.

IoT4CPS tools should support dependability in several levels of CPS hierarchy. Our encryption solutions would enable efficient and secure communication channel between devices with limited resources, which is the case in a typical IoT scenario. In extreme cases of devices uncapable of encryption, we can at least guarantee data authenticity by applying digital watermarking techniques. IoT recommender system further enhances dependability by guiding the user to select the appropriate protocols, thus making the entire system more reliable. On a platform level, we increase the trust in system by leveraging methods for trusted orientation and localization. A system which adopts our SHSA techniques gains the ability to manage unpredictable component failures, thus becoming more dependable. Finally, our design-time security and safety tools, V&V patterns and Cryptographic API development guidelines aim to increase dependability early on.

# 4. Application Layer Tools and Methods

In this section we focus on IoT4CPS WP3 contributions for improving dependability on the application level. We report on our design time security analysis tools for model-based devilment, for security requirement management and threat modelling as well as safety and security risk assessment.

## 4.1 GSFlow

Our first contribution is a tool for standard-based product development management: GSFlow. It is one of the results of a more general effort to develop tools to support model-based development approaches and safety & security by design. The goal of GSFlow is to support the complete engineering lifecycle of safety and/or security relevant systems based on pre-defined processes, by guiding the user through the development process. Its main objective is to make standard driven development straightforward, especially for companies that are unacquainted with functional safety and security standards. The model implemented in GSFlow simplifies standard driven development by guiding the end-users through the development process and consequently offloads the effort from security experts, while still providing assurance. This is especially relevant to SMEs which only have a limited number of safety or security experts.

The central output of GSFlow is a safety and/or security case which shall support companies two ways: (1) helping to reach assurance for their products and (2) allowing to summarize the argumentation why a system is acceptably safe and secure. To achieve this goal every project in GSFlow contains requirements which correspond to functional safety and security standards such as [37]. GSFlow provides standard conformant user management and utilizes tools to ensure the quality of an evidence.

GSFlow provides a flexible framework for modelling and executing standards. It is also capable of executing plugins written by an external developer. This flexibility ensures that the specific needs of a company can be met. Furthermore, an external developer is only required to implement an interface according to their needs. For example, GSFlow can execute external plugins to generate reports, as well as to check the artefacts and requirements using Natural Language Processing methods.

### 4.1.1 GSFlow Structure and Definitions

In this section we provide more details about the operation of GSFlow. Requirements are defined as the entities needed to achieve the objective of the project. Two different kinds of requirements can be distinguished. The first kind of requirements are the standard requirements, which are derived from functional safety and security standards. They are needed to identify the goals that are obligatory to reach compliance to relevant standards. Secondly, the product requirements in GSFlow represent the requirements that are specific to a product. They can be linked to a standard requirement. Furthermore, the requirements in GSFlow are the key elements for documenting the workflow as they serve as an anchor to attach evidence and trace every action that is conducted on them while being processed. Once processing is completed and all evidence has been created, a requirement may enter the state of completion.

Phases in GSFlow represent phases of a safety/security standard. GSFlow ensures that in every phase Standard Requirements that are specific to this phase need to be fulfilled. A phase can only start once all previous phases have been completed. Phases in GSFlow are used to structure requirements and define the order of execution in the workflow.

GSFlow allow for integration with external tools. By implementing and using the interfaces provided by GSFlow, an external Developer can create plugins and an admin can upload them to GSFlow after conducting validation. These tools/plugins are then ready to be executed. The dataflow between GSFlow and the tool/plugin is conducted only through the API. The utilization of tools shall serve as quality assurance as well as provide convenience to the users working with GSFlow.

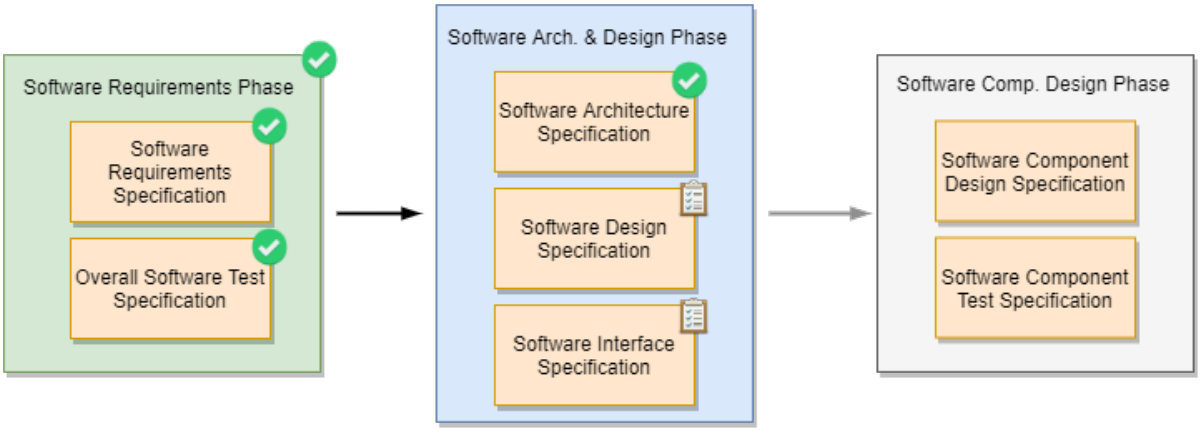


**Figure 4: Illustration of workflow and fulfilment status**

GSFlow supports standard conformant user management. Different roles can be assigned to different users per project. Roles are based on a generic model of roles defined in different standards. In GSFlow, roles can be mapped to specific standards. The list of the supported roles includes Project Manager, Requirements Manager, Developer, Verifier, Validator & Assessor.

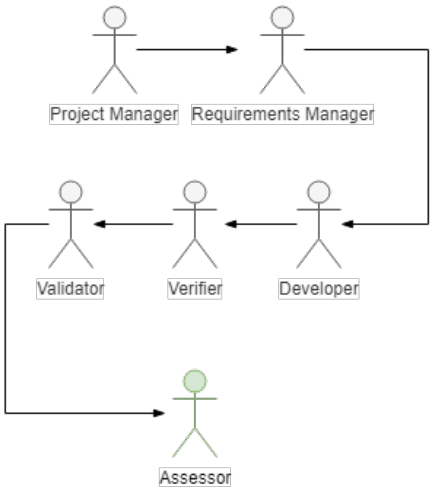


**Figure 5: Basic responsibility chain in GSFlow**

In GSFlow responsibilities are modelled on a per requirement basis. Each requirement has the required roles assigned to support the development chain defined in a standard. To be more precise, when the developer has finished their task, the verifier is assigned followed by the validator. Only when all the responsible parties have marked their tasks as finished, assessment can take place.

As discussed, GSFlow enables the end user to organize their workflow according to phases, work products and requirements deducted from standards. GSFlow also serves as a documentation platform and tracks every action that is conducted regarding a certain requirement and links evidences to those requirements. This way, GSFlow supports traceability. The flexible framework inside GSFlow enables tailoring of generic processes to company or project specific demands. When appropriately modelled, it can support a safety and security co-engineering workflow. To achieve this, different adequate standards need to be analysed and consequently modelled into one combined workflow.

For example, the standards ISO/IEC 27002[35] and EN 50128 [34] contain requirements and methods emphasizing on availability, reliability, confidentiality, integrity and maintainability. These standards describe measures that need to be undertaken in order to assure the attributes. SAE Standard J3061 [37], which is a Cybersecurity Guidebook for Cyber-Physical Vehicle Systems, describes security and its effects on safety on financial and operational basis and hence, availability and reliability. Security and measures to ensure integrity

and confidentiality can be found in the IEC 62443 series [38]. By following the generated workflow, GSFlow makes the development of dependable IoT systems feasible.

## 4.2 Failure Mode, Vulnerabilities and Effects Analysis

Failure Mode, Vulnerabilities and Effects Analysis (FMVEA) is a static method for security analysis. FMVEA is based on the Failure Mode and Effect Analysis (FMEA) and extends the standard approach with security related threat modes [23]. A *failure mode* describes the way the function of an element fails. A *threat mode* describes the way in which the identified function of an element can be misused. Threat modes classifies threats in six categories (Spoofing of user identity, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege).

FMVEA consists of several phases, beginning with system modelling phase. Once this phase is complete the failure and threat modes for each element of the system model are identified. Depending on the domain, the system architecture and the knowledge about the system, failure and threat modes can be refined and extended. Each identified failure or threat mode associated with the element is investigated for potential effects. For modes with critical effects, potential causes are analysed and the likelihood for each cause is estimated. For threat modes, likelihood is determined using a combination of threat and system properties.

Threat properties mainly describe the resource and motivation of a potential threat agent while system properties include reachability and system architecture. The system model is based on a three-level data flow diagram (DFD). Effects of failure and threat modes are presented at the context level of the diagram, which shows the interaction between the system and its environment. Failure and threat modes are located at the level 1 DFD. Vulnerabilities and failure causes are based on the level 2 DFDs.

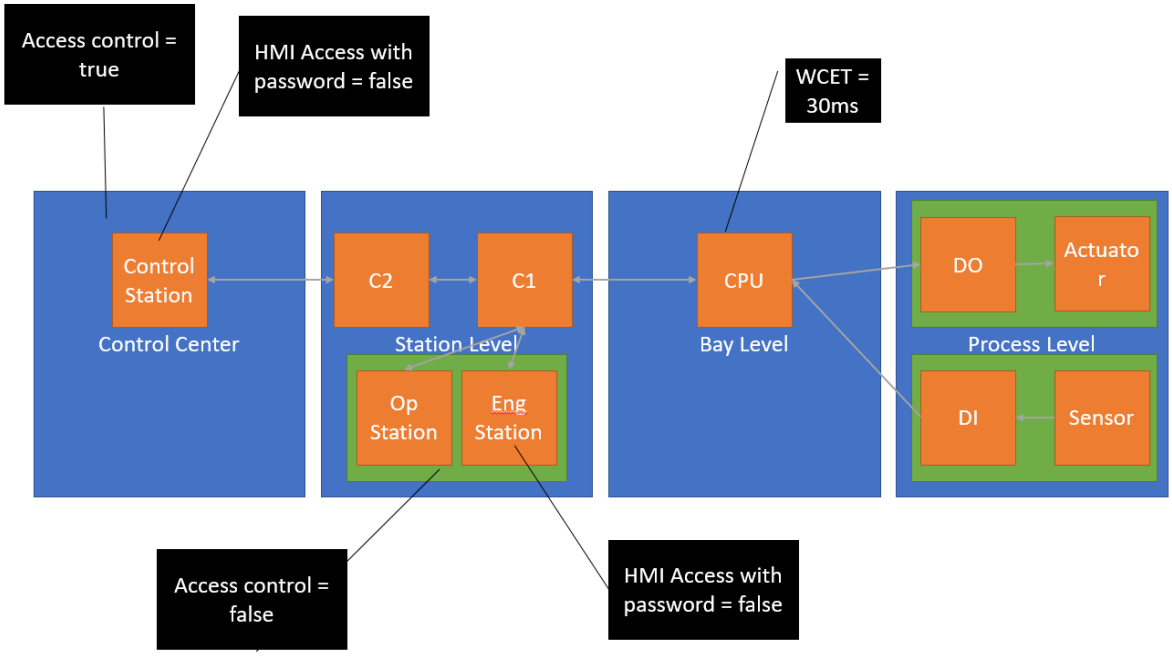


**Figure 6: An example model as an input for analysis with FMVEA**

Figure 6 shows the diagram of an example use-case in which a set of actors is controlled by an CPU, based on some sensor input. Control Strategy can be defined and monitored in some higher layers. The example model will be used to perform an analysis with FMVEA. The blue squares represent the root environments of a specific level. Each orange node can be seen as an operating element inside the environment. Green squares represent sub-environments inside the root-environments which encapsulate their own operating elements. Black squares display the defined attributes/properties for the diagram elements. For example, the “CPU” has the attribute/property “WCET” (“Worst Execution Time”) with a value of “30 ms”.

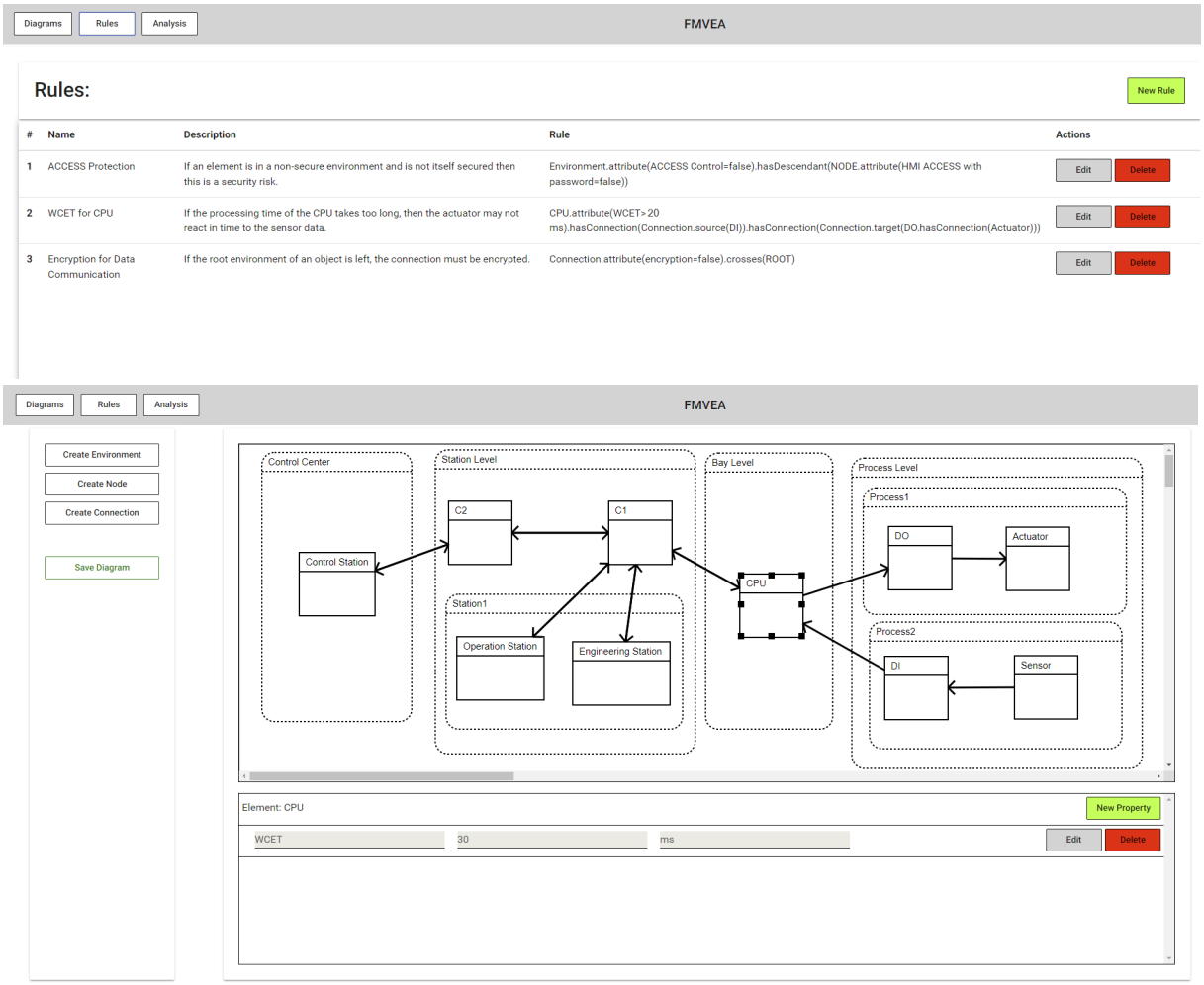


**Figure 7: The diagram modelled in FMVEA and the rules**

Figure 7 shows the diagram from Figure 6 modelled in FMVEA. On the left-hand side, we can see the diagram related actions like “Create Environment”, “Create Node” and “Create Node”. An Environment can be considered as a container, which provides general attributes to his children. Attributes can be focused on Security and Safety. The attributes of an element are directly displayed below the diagram. Figure 7 also displays the rules which should be used to analyse the use-case diagram. From the left to the right are the names of the rule then a short description and the “Rule”. The “Rule”-column is the most important one, because here the actual rule for the analyser is defined. The previously created rules are applied on the selected diagram.

From left to right you one can observe the applied rule and the results of the specific rule on the diagram. The affected elements and connections can be viewed in the diagram if the user clicks on the “Show”-button to the right. Inside the diagram the affected elements and connections get highlighted by a red border as you can see in the Figures 11-13.

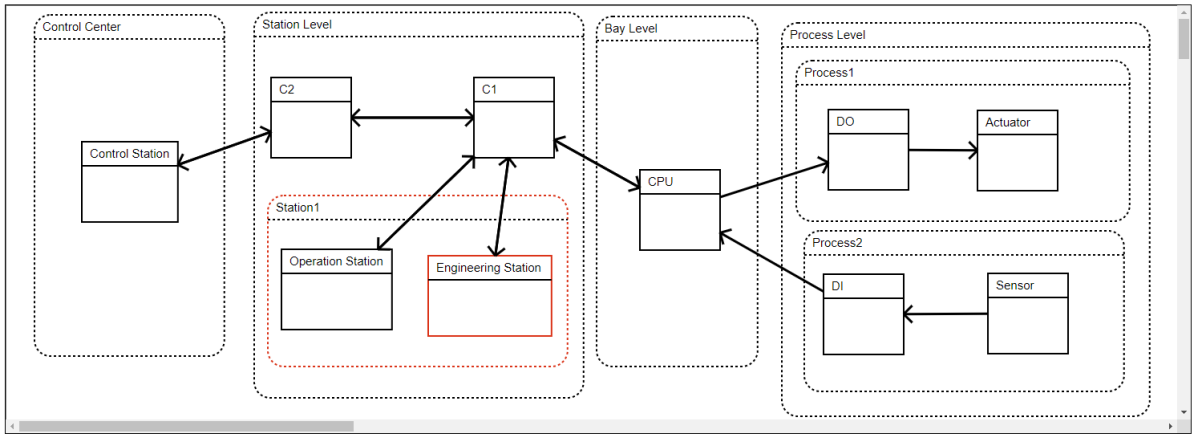


**Figure 8: 1st rule results**

In Figure 8 you can see the affected elements of the first rule. The Definition of the first rule says that if there is any environment with the property “ACCESS Control=false” which contains a child object with the property “HMI ACCESS with password=false” then there is a security problem. As one can observe from Figure 6 we initially defined the “Control Station” with “ACCESS Control=true” so there is no problem even if the “Control Station” has the property “HMI ACCESS with password=false”. However, if one observes the “Station 1” and the “Engineering Station” then both criteria are fulfilled, and this is the reason why these two objects are the affected objects of the first rule.

In Figure 9 you can see the affected connections of the second rule. The Definition of the second rule says that if there is any connection between two objects which do not share the same root object then the connection must be encrypted. One can observe from Figure 6 we never defined an encryption property for any connection inside the diagram. For example, take the connection between the “Control Station” and “C2”. The root object of the “Control Station” is the “Control Center” and the root object of the “C2” is the “Station Level” but they share a connection. Now let’s consider the connection between the “C2” and “C1”. The root object of both objects is the “Station Level”. They share the same root element as the same parent element, therefore this connection does not represent a potential problem.

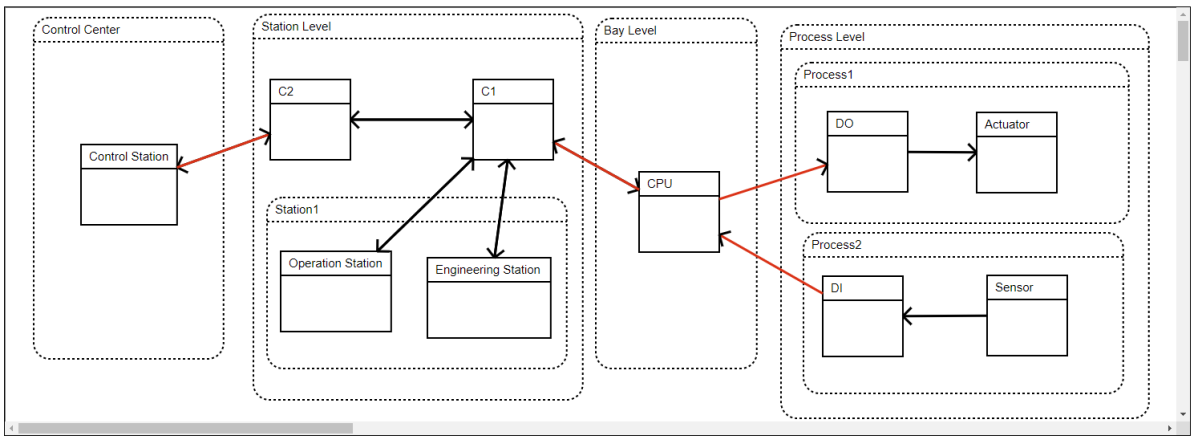


**Figure 9: 2nd rule results**

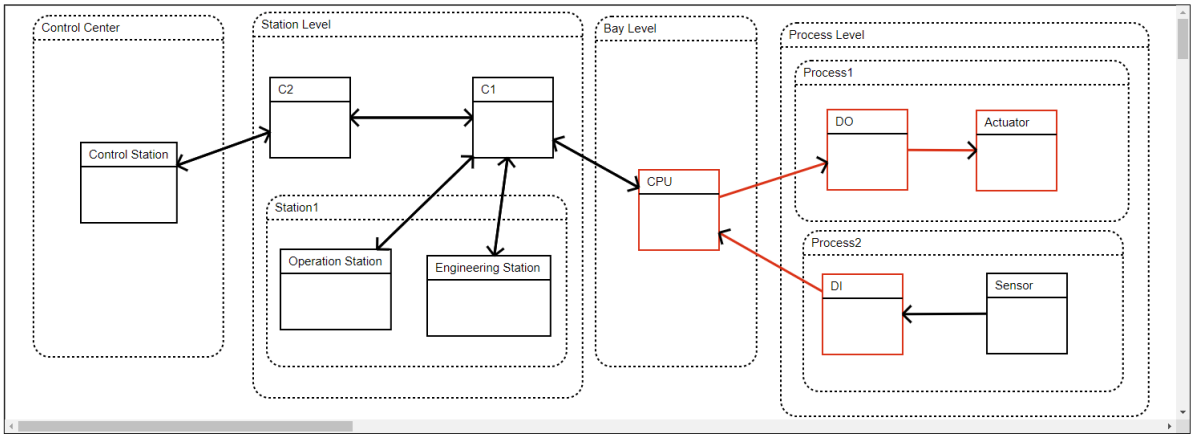


**Figure 10: 3rd rule results**

In Figure 10 you can see the affected elements and connections of the third rule. As you can take from Figure 6 we set the WCET of the CPU as “30 ms”. The Definition of the third rule says that if the WCET inside the CPU is higher than “20 ms” the actuator may not react in time to the sensor data. The sensor data is coming from the “Data Input” (DI) then the data is processed inside the CPU with a WCET of 30ms and then sent to the “Actuator” over the “Data Output” (DO). This is the reason why the “DI”, “CPU”, “DO” and the “Actuator” are affected elements in this case. The connections between these objects transport the data and are affected connections in this case. This rule could be expanded by additionally considering the latency of the connections.

## 4.3 Threat Modelling

A secure system can be designed and developed only if security issues are well-identified and addressed appropriately in the early stages of the system development. That is considered a significant advantage because once the system is developed, it becomes harder to add security countermeasures. ThreatGet is a toolbox for Enterprise Architect, which is a widely used tool for Model-Based Systems Engineering. ThreatGet identifies, detects, and understands potential threats in the foundation level of system models. It supports the initial steps of the developing system process to guarantee the security by design. The following figure depicts an example design which a user can specify in ThreatGet, in order to perform security threat analysis.

Our Figure 11: IoT-based Smart Factory illustrates an example of IoT application in a smart factory, which is one of two main use cases of IoT4CPS project. The example contains one or more sensors and camera units for collecting and gathering information about the production line. The collected data is processed by a control unit, that handle and manage actions by sending signals to actuator units such as robotic arm and engine as defined in the figure. The data is sent to a centralized data storage and processing unit for monitoring the quality of the production.

In such a heterogeneous and distributed system, potential security threats can exist in any component, which may compromise the operation of the entire system. In order to identify potential threats in our system, we apply ThreatGet. The threats are defined according to the dataflow from the source components to the targets. According to the security properties of these units, some vulnerabilities could be exploited be threats.

In Figure 11 the Control Unit takes the central role in the Smart Factory model. This component communicates with the data storage through a gateway, which runs a certain communication protocol. Depending on the gateway device, it can provide low or high security features. In our model, we can analyze devices with different levels of security by adjusting the possible security parameters of the model. We model possible security mitigation measures in the communication flow between the Control Unit and the Gateway. The features

include: Source and Destination Authentication, Confidentiality and Integrity. We can set these parameters to values which correspond to a real device. In Figure 12 we can observe that if the HTTP protocol has mitigation measures switched off, the communication channel introduces 6 different threats. Once we introduce the mitigation measures in the communication channel the number of threats reduces to 4.

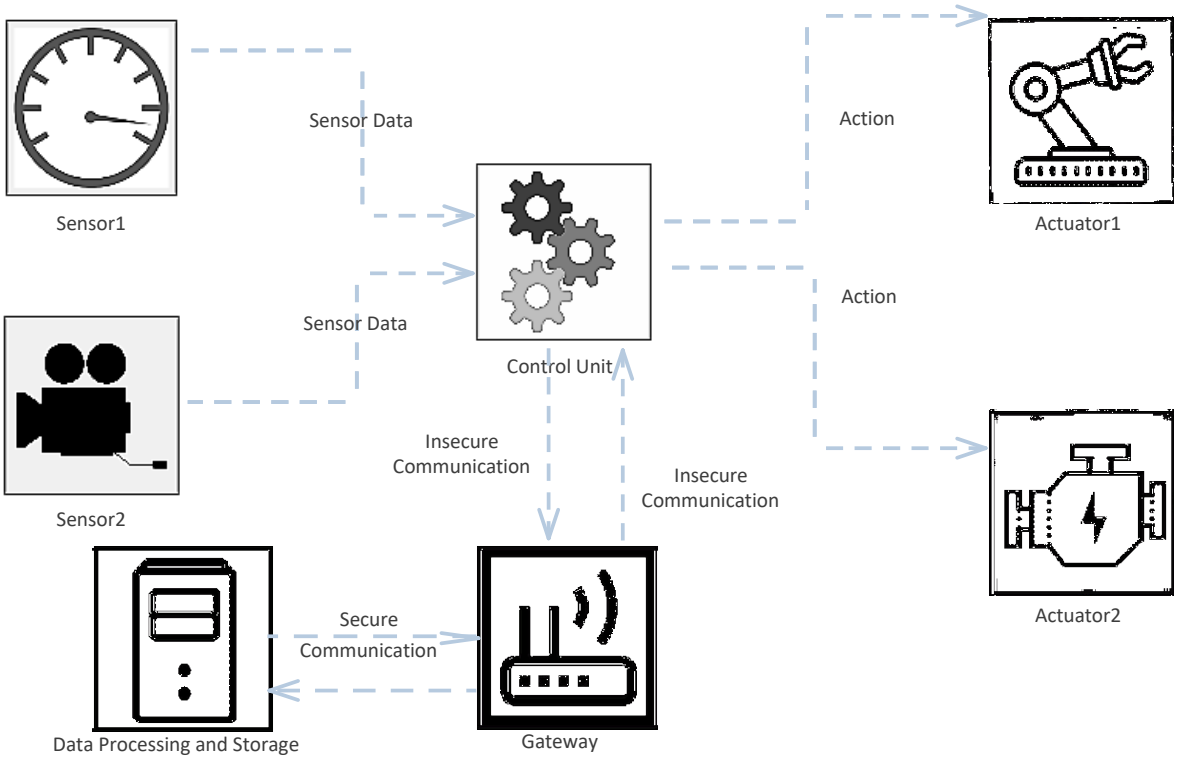


**Figure 11: IoT-based Smart Factory**

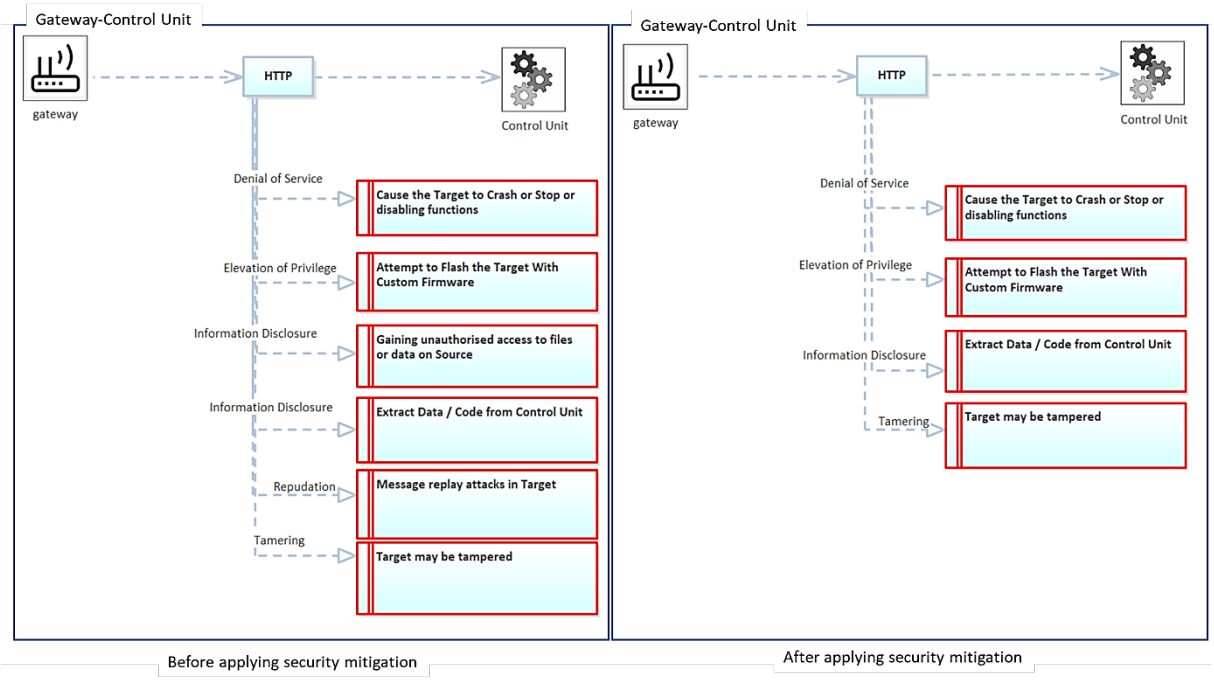


**Figure 12: The impact of introducing a mitigation measure**

As a result of the analysis, ThreatGet detects 32 potential threats, that are classified according to the STRIDE model (i.e., Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service (DoS), and Elevation of Privilege). Spoofing represents an attempt by a person or program to identify itself as another by falsifying data, to gain an illegitimate advantage. Data tampering is an attempt to maliciously modify the data through unauthorized channels. Repudiation is a kind of attack which manipulates the log data in the computer systems, in order to conceal traces in the log. Denial of Service and Elevation of privilege are well-studied threats, where an attacker is jamming the access to the system resource, and when an attacker attempts to gain more access rights than allowed, respectively.

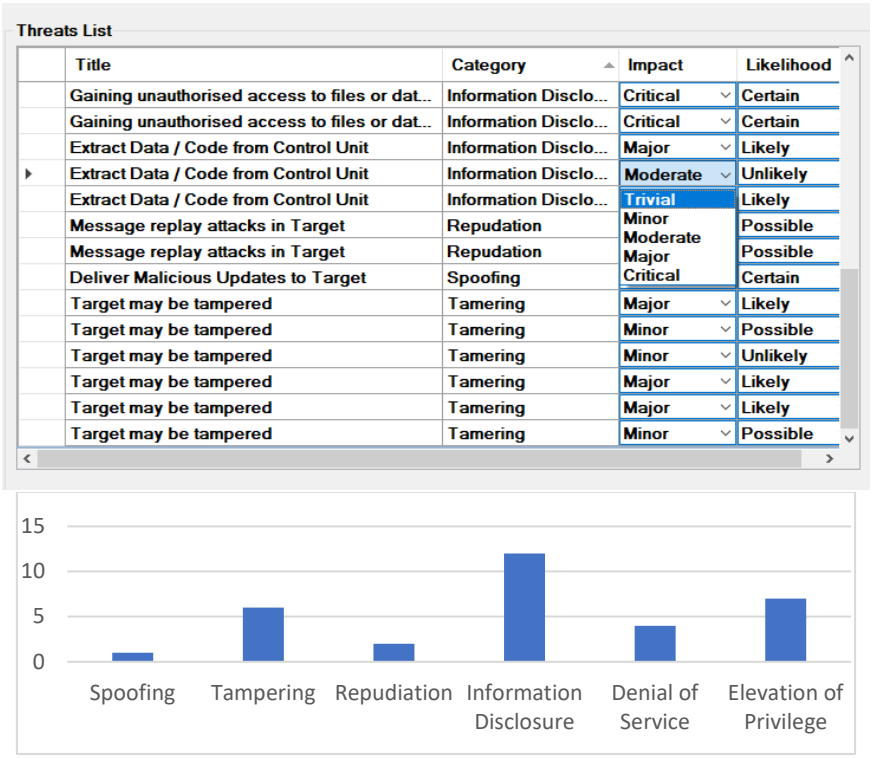


**Figure 13: Distribution of identified threats according to the STRIDE model and all potential threats listed**

Afterwards, the risk assessment process is applied to evaluate risk severities of the detected threats according to the values of the impact and likelihood parameters. The ThreatGet's user can select the parameters of the impact and likelihood to estimate the risk severity, as shown in Figure 13. There are five parameter values of the Impact and Likelihood are used in the estimation process to determine the risk severity level of threats. Figure 14 illustrates the levels of the impact and likelihood that are used by ThreatGet for the risk assessment process.

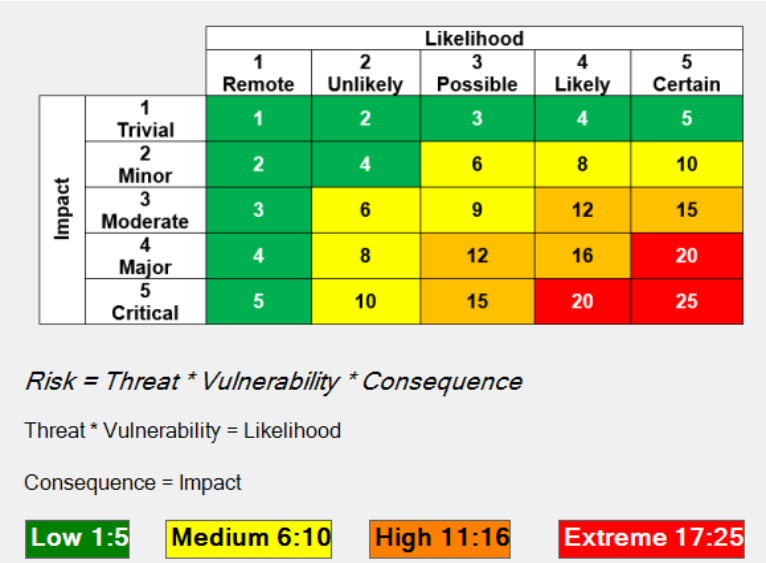


**Figure 14: Risk assessment chart**

The next step is to address the identified potential threats by the most applicable security countermeasure for mitigating the overall risk of the predefined system model. Furthermore, we use IEC 62443 as the most suitable security standard to provide a flexible framework for addressing current and future security vulnerabilities could be exploited by potential threats [29]. In this case we should define two main values that could help to reach the high level of security protection. The first one is called the Security Achieved (SA), this value defines the current security level that is achieved after applying security countermeasures against security vulnerabilities. The second value is to define the actual security goal that the system architect wants to be achieved this parameter is called Security Target (SA) [28].

In this example we claim that we are looking forward to adding all the threats with the highest severity level. Furthermore, the SA = 7 (the total number of extreme potential threats in this example), and SA = 0 because there are no security countermeasures are applied yet. The following part discuss the security treatment process for addressing the security threats by security countermeasures.

### 4.3.1 Risk Treatment for the Identified Extreme Threats

This process plays an important role to address potential threats with security countermeasures that should be considered in the system developing phases to keep the risk always low. The system security target (ST) is defined to estimate the maximum-security level needs to be achieved. Also, the current status of the system security level is defined as the security achieved (SA).

In this example, we are looking forward to reducing the security level of the unacceptable risk level. Furthermore, we focus on the Extreme severity level of threats (unacceptable risk level) that need to be addressed by proper security countermeasure. Thus, ST is seven (the total number of extreme severity risk), and SA is one until addressing these threats. The value of SA is incremental according to change in security level after applying additional security countermeasures. This process is completed if SA = ST; otherwise, additional security countermeasures should be applied for security improvement. Figure 15 illustrates the difference between the current and the target security levels before applying security countermeasure.

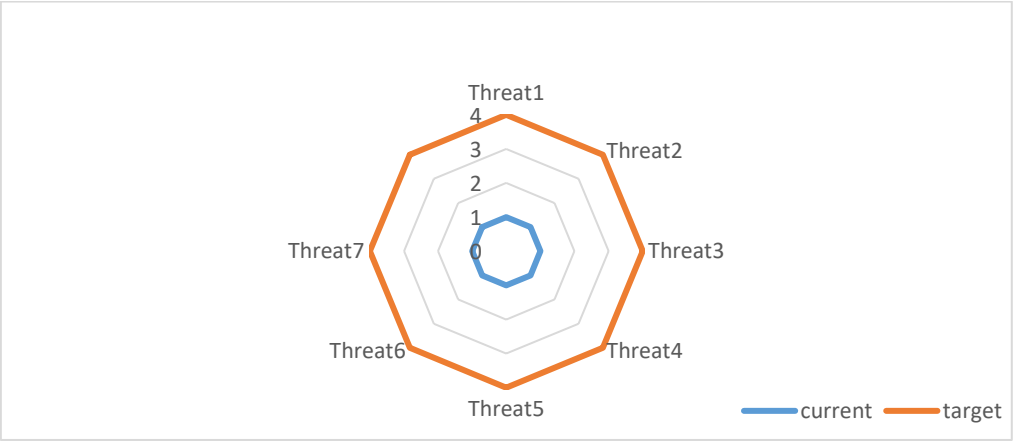


**Figure 15 The current (blue) and target (orange) security levels before applying security countermeasures**

To convey the ThreatGet workflow, we proceed with handling security issues for the most prominent threats of our model, and with showing the outcome of applying a mitigation measure. The following sections define the selected security countermeasure based on IEC 62443 part 4-2 for addressing the existing potential threats.

### 4.3.2 Risk Treatment for Threat 1 and Threat 2

***Threat1 Title: Message replay attacks in Target***
***Threat1 Category:*** *Information Disclosure*
***Threat1 Origin:***

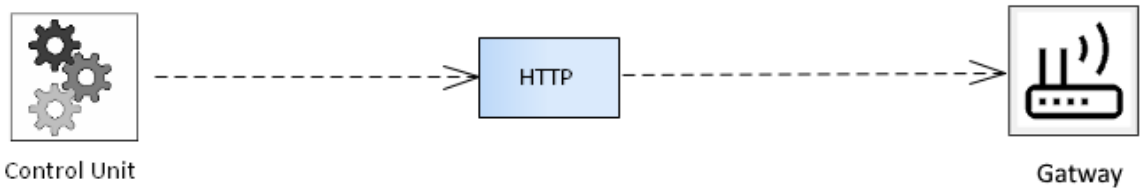


**Figure 16 Threats 1 and 2 propagation source**

The following table shows a list of selected security countermeasures for addressing this threat. These security countermeasures should be applied to mitigate the threat. The following abbreviations are used: CR: Component Requirement, NDR: Network Device Requirements, and EDR: Embedded Device Requirements.

| IEC62443 Classification | Name of the security requirement |
|---|---|
| **CR 2.11** | Timestamps |
| **CR 2.10** | Response to audit 48 processing failures |
| **CR 2.12** | Non-repudiation |
| **NDR 3.14** | Integrity of the boot process |
| **EDR 3.14** | Integrity of the boot process |

**Table 1 IEC62443 Requirements for Threat1**

***Threat2 Title: Target may be tampered***
***Threat2 Category:*** *Tampering*
***Threat2 Origin:*** *shown in Figure 16*
The following table defines a list of the chosen security countermeasures for addressing threat2.

| IEC62443 Classification | Name of the security requirement |
|---|---|
| **NDR 3.11** | Physical tamper resistance and detection |
| **CR 1.9** | Strength of public key-based authentication |
| **NDR 3.14** | Integrity of the boot process |
| **EDR 3.11** | Physical tamper resistance and detection |
| **EDR 3.14** | Integrity of the boot process |

**Table 2 IEC62443 Requirements for Threats 3&4**

### 4.3.3 Risk Treatment for Threat 3 and Threat 4

***Threat Title: Gaining unauthorized access to files or data on Source***
***Threat Category:*** *Information Disclosure*
***Threat Origin:***

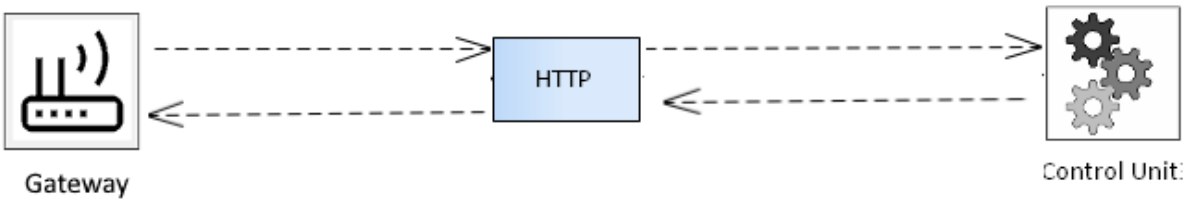


**Figure 17 Threats 3 and 4 propagation source**

The following table shows the chosen security requirements for these threats.

| IEC62443 Classification | Name of the security requirement |
|---|---|
| **CR 4.1** | Information confidentiality |
| **CR 4.3** | Use of cryptography |
| **CR 3.1** | Communication integrity |
| **EDR 3.2** | Protection from malicious code |
| **NDR 3.2** | Protection from malicious code |
| **CR 3.8** | Session integrity |

**Table 2 IEC62443 Requirements for Threats 3&4**

## 4.4 MORETO

The correct security requirement identification and efficient security requirement management are essential for any security engineering process. We can design, implement, and test a secure system only if we know the exact security requirements. Achieving an efficient requirement management is a challenge in system development. The Model-based Security Requirement Management Tool (MORETO) serves a tool for security requirements analysis, allocation, and management using modelling languages such as SysML/UML. MORETO is an Enterprise Architect (EA) plugin for managing the IEC 62443 security standard. It is a reliable and a flexible to model safety & security requirements suited to different components and system architectures. It generates a list of security requirements in a given diagram, which can help the user to build-up a secure infrastructure. Figure 18 shows a simple example of different components which interact together through a network.

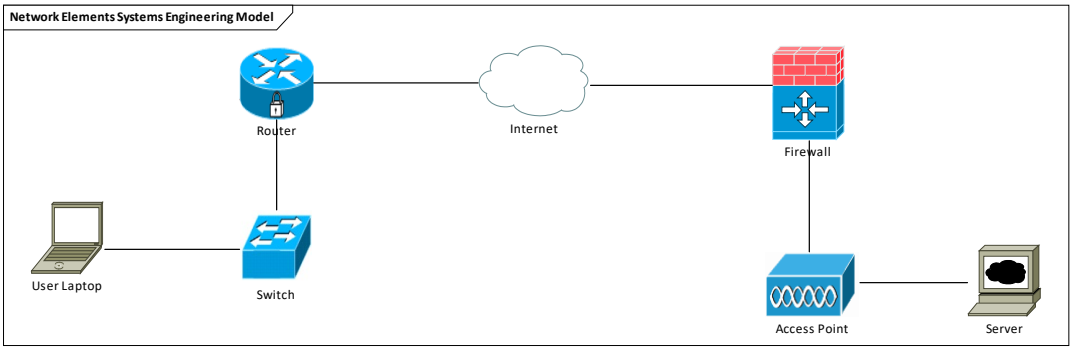


**Figure 18 Simple network elements for system engineering model by MORETO**

MORETO scans all the elements of a given model and automatically generates a list of security requirements based on expert knowledge encoded in the MORETO knowledge base itself and on the description contents of the IEC 62443. Figure 19 shows a list of identified security requirements of the Router and Switch devices respectively.

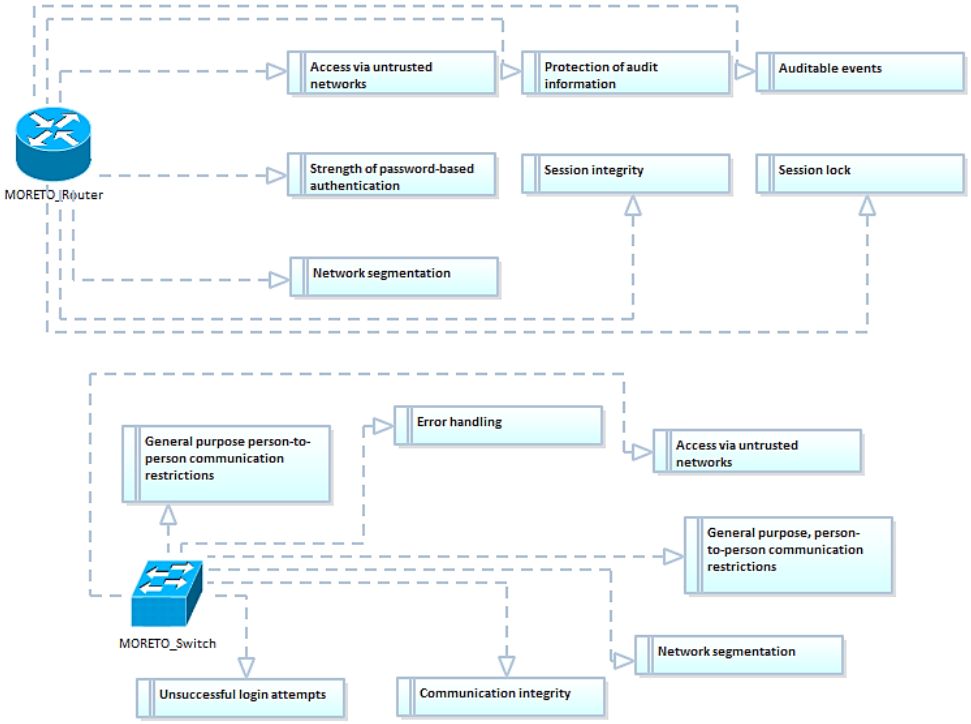


**Figure 19 IEC 62443 Security standards for router and switch devices**

## 4.5 Safety and Security co-engineering

In order to build dependable IoT systems, it is insufficient to regard concepts of safety and security separately since safety and security of IoT are not independent. Therefore, we propose a combined risk assessment approach for safety and security and demonstrate it using tools developed in IoT4CPS project.

The state-of-the-art approaches in Threat Analysis and Risk Assessment (TARA) such as Microsoft STRIDE and FMVEA provide a risk assessment approach but they do not consider multiple attacks (attack combinations via 'OR', 'AND', 'SAND' (sequential AND)) and multi-stage attacks. On the other hand, Attack Tree Analysis (ATA) approach considers such attacks but either in a purely qualitative or purely quantitative fashion (which is not feasible for most applications). There is no TARA-based approach which considers how the likelihood (final risk) can be evaluated in case of multiple attack combinations and multi-stage attacks.

The security risk assessment is performed combined with ADTool [30], which provides a simple way to create and edit attack trees. Attack trees are conceptual diagrams showing how an asset, or target, might be attacked. Attack trees have been used in a variety of applications. The attack tree is then exported as an XML file to the scripts [32] which implement the risk assessment algorithm. During evaluation two inputs are required by the user: the threat level scores for all the BAS (Basic Attack Steps) and Impact level scores for all the Higher Attack States.

### 4.5.1 Security Risk Assessment with Attack Trees

The generation of attack trees, which is a prerequisite for Attack Tree Analysis, is presently not covered in our work and is a subject of future work. The approaches such as Microsoft STRIDE generate general threats corresponding to STRIDE category by a predefined mapping of these threats to general elements of a system (approach represents almost all systems with four general elements). Therefore, the STRIDE and Data Flow Diagram (DFD) approach are sufficient to generate threats for each element. However, to generate an attack tree we also need to know how these threats affect each other, which can be learned from security attack databases. In case of automotive, AUTO-ISAC (SAE Vehicle Electrical System Security Committee) or public Automotive Attack Database [31] can be considered as a relevant data source. The concrete specification of system design is not known in the early stages of design process, which renders the very large list of all the possible attacks, thus making the STRIDE approach preferable. However, in the later part of the design phases specific attacks can be used to generate attack tree.

We propose a Security Risk assessment via attack trees, for evaluation of logical combination of attacks, based on the following principles:

a) For considering OR combination of attacks, we will evaluate the canonical form (DNF-Disjunctive Normal Form) of the attack tree. In canonical form we get all the attack paths related with OR (disjunction) relation, to an attack state.
b) Each attack path consists of several basic attack steps related with AND (conjunction). This means the attack path will be successful if and only if all the basic attack steps involved are successful.
c) For considering AND combination among basic attack steps, we evaluate the final likelihood as same as the lowest of the threat level, as this is the critical threat and system cannot be compromised until this threat succeeds.
d) For considering OR combination, the canonical form gives all attack paths to each state related with OR (disjunction) and we do the risk assessment for each attack paths individually.
e) For considering Sequential AND attacks, same as AND, we did not consider impact of sequential dependence on threat level (likelihood). The reachability score for following attacks can be taken as preceding attack, as this provides a gateway to the following attacks.

### 4.5.2 An example of Security Risk Assessment with Attack Trees

The Attack Tree shown in Figure 20 has been constructed (by our judgement for possible attack scenario) from the threats listed in [40]. Since the attack tree lists specific attacks instead of generalized threats (STRIDE category), the analysis is performed in a later phase of the design process, and basic security strategies have been already adopted to secure system from basic possible attacks.

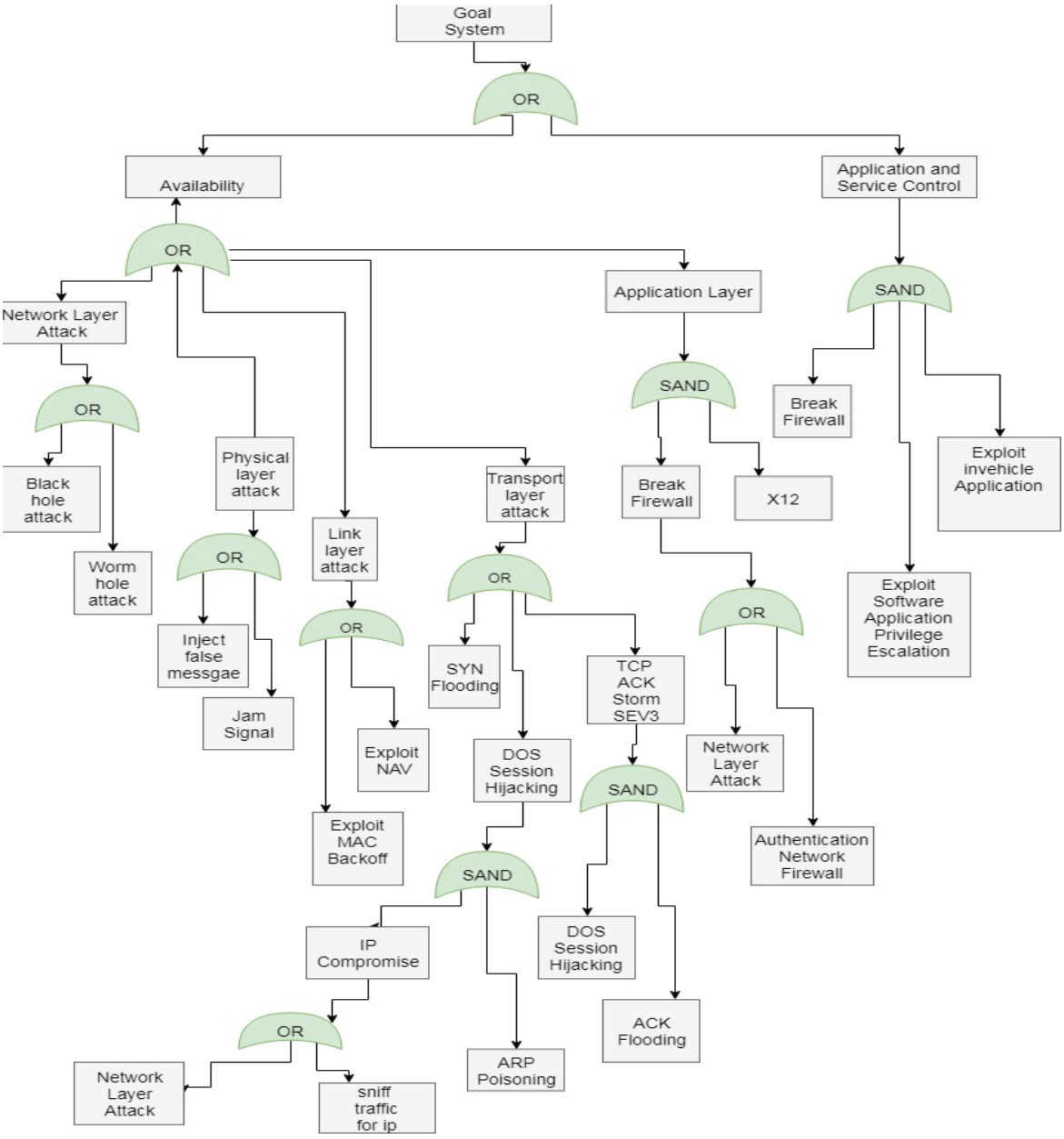


**Figure 20 An attack tree for Security Risk Assessment**

**Steps for risk assessment using the ADtool**

**Step1** – The attack tree is drawn with help of ADTool, as shown in Figure 21.

**Step2** – The model is imported to VBA, Excel, as an .xml file. The VBA scripts sorts basic attack steps (which are at bottom of attack tree) and higher attack states.

**Step3** – The evaluator needs to provide scores for threat level parameters and Impact level.

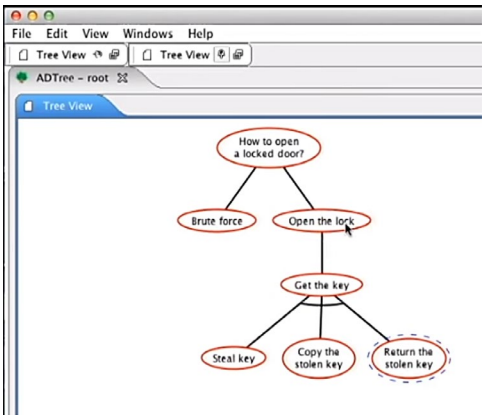


**Figure 21 An attack tree obtained using ADT tool**

**Step4** – The VBA scripts then evaluate, for all higher attacks, which are the basic attack steps involved and how they are related using 'AND', 'OR', 'SAND', as shown in Figure 22.

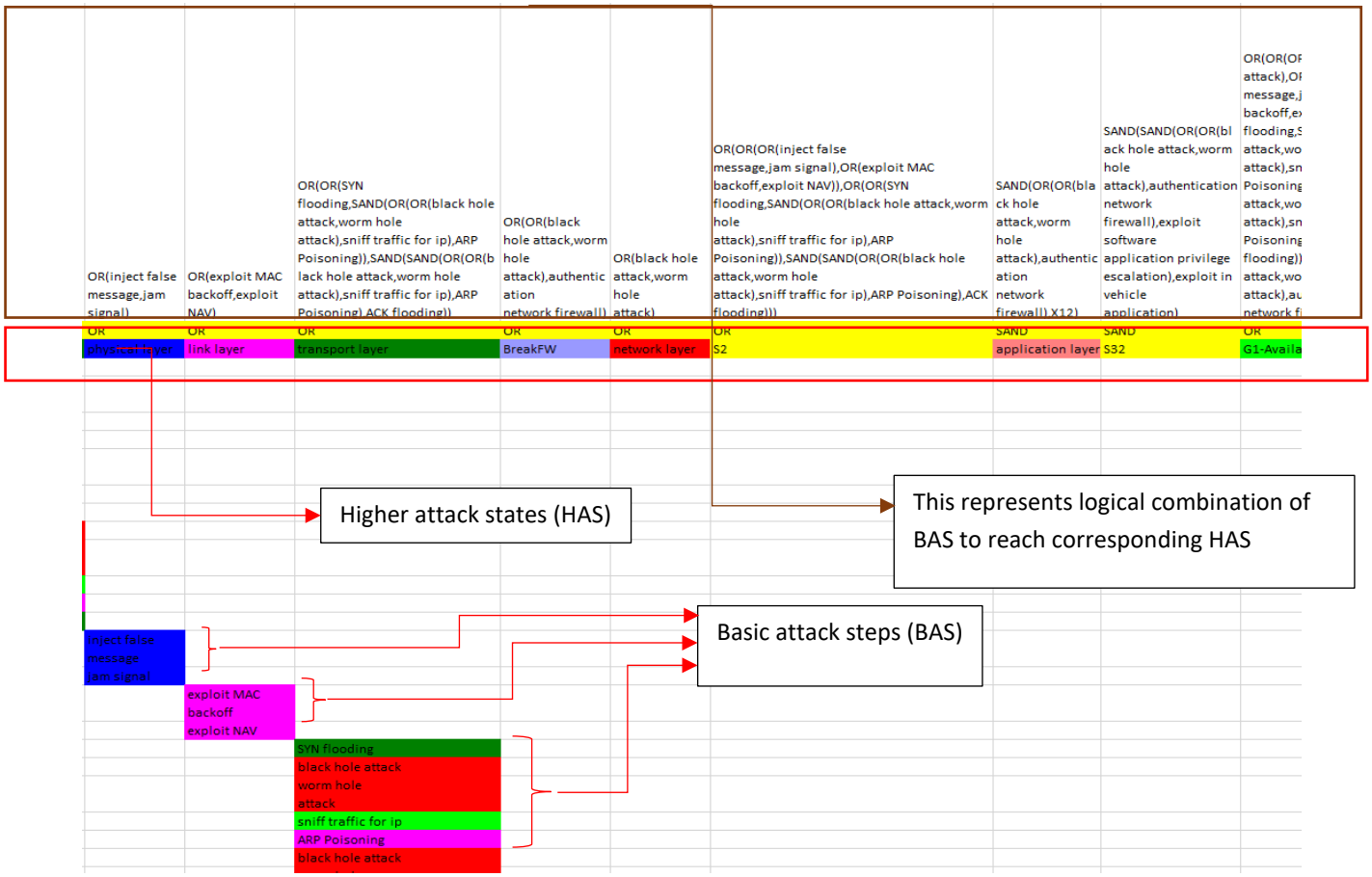


**Figure 22 Basic attack steps and relations**

**Step5** – The implemented scripts evaluate canonical form (disjunctive normal form) for all higher attack states. For example, the higher attack state 'G1' consist of three attack paths, these attack paths are related to each other with disjunction (HAS will be compromised if any of the three attack paths are successful). The basic attack steps inside each attack path are related with conjunction or sequential conjunction, i.e. attack path will succeed only if all the basic attacks are successful.

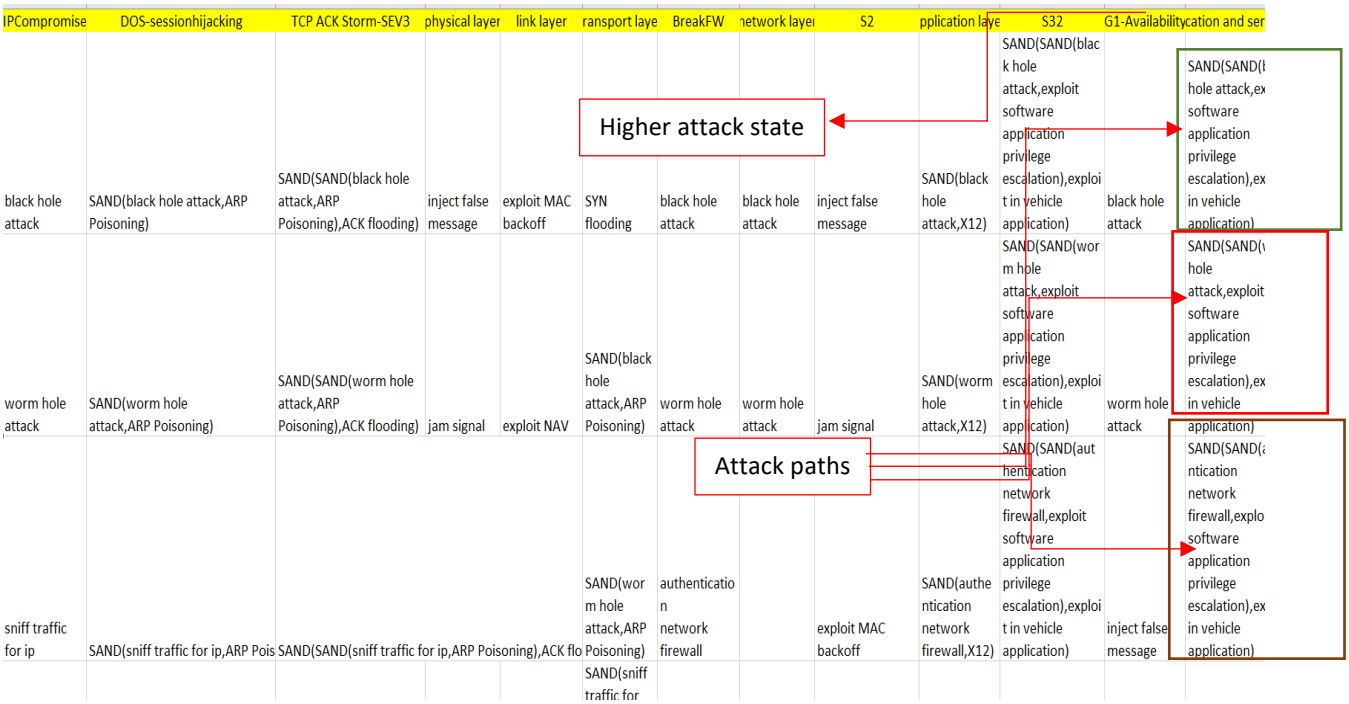


**Figure 23 Disjunctive Normal Form for all higher attack states**

**Step 6** – Considering strategy listed in section 7.1, the critical basic attack and its likelihood will be evaluated for each attack path. Finally, the risk matrix will be obtained using *risk = likelihood x impact*, shown in Figure 24.

| | IPCompromise | DOS-sessionhijacking | TCP ACK Storm-SEV3 | physical layer | link layer | transport layer | BreakFW | network layer | S2 | application layer | S32 | G1-Availability | Application and service control | GOAL-SYSTEM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| sniff traffic for ip | 6 | 12 | 18 | | | 6 | | | 6 | | | 6 | | 6 |
| ARP Poisoning | | 18 | | | | 9 | | | 9 | | | 9 | | 9 |
| ACK flooding | | | 21 | | | 7 | | | 7 | | | 7 | | 7 |
| inject false message | | | | 16 | | | | | 8 | | | 8 | | 8 |
| jam signal | | | | 12 | | | | | 6 | | | 6 | | 6 |
| exploit MAC backoff | | | | | 16 | | | | 8 | | | 8 | | 8 |
| exploit NAV | | | | | 16 | | | | 8 | | | 8 | | 8 |
| SYN flooding | | | | | | 9 | | | 9 | | | 9 | | 9 |
| authentication network firewall | | | | | | | 14 | | | | | | | |
| black hole attack | 11 | | | | | | 22 | 22 | | | | 11 | | 11 |
| worm hole attack | 12 | | | | | | 24 | 24 | | | | 12 | | 12 |
| X12 | | | | | | | | | | 21 | | 7 | | 7 |
| exploit software application privilege escalation | | | | | | | | | | | | | | |
| exploit in vehicle application | | | | | | | | | | | 24 | | 6 | 6 |

**Figure 24 Calculated Risk Matrix**

The entries of the matrix represent the risk scores, the row entry tells us which basic attack step is critical, and column entry shows for which higher attack state.

### 4.5.3 Safety Risk Assessment and Combined Risk Assessment Approach

There is a number of safety risk assessment methodologies, each having certain benefits and certain limitations. For example, the limitation of the Failure Mode and Effect Analysis is that it is more suited to analyzing systems which have no redundancy and no multiple failures. Fault Tree Analysis (FTA) on the other hand allows logical

combination of events but is generally qualitative, can be very complex to be solved analytically and doesn't support repair events. Continuous Time Markov analysis supports repair events but can cause state explosion problem for complex systems and supports only exponential probability distribution functions. Therefore, hybrid complex systems such as standby switching systems, where the main system, switching system and standby system each has different failure distribution, it is very complex to solve analytically. In such cases Stochastic Colored Petri Net (SCPN) models with simulation are used, they also take care of state explosion problem. Therefore, for redundant systems and complex dynamic systems (switching systems) we should use SCPN model and simulation, such as TimeNET tool.

A complex system however can use individual benefits of each of the methodologies. Such as the highly redundant and complex part of the system can be independently analyzed with SCPN, and the equivalent system can be replaced by a single element with corresponding failure rate. The modified system can be analyzed with the help of FTA.

The identification of the Target of Evaluation (TOE) for threat identification in security evaluation is entirely different from TOE for failure analysis, due to a very different kind of grouping of assets. Now when we consider cross domain impacts e.g. if some security attack X is contributing to some failure mode A, we need to then trace how this failure mode is propagating to the different systems of interest which are safety critical.

At every design iteration, we must consider dependence among the attributes, e.g. how security is impacting Safety requirement of the system, how safety or security is impacting performance, so that when we are trying to make design decisions, we must know that TOE/Asset should satisfy requirement w.r.t all the attributes.

Therefore, if we have a lot of cross domain impacts the tracing of failure (induced by security breach) to all the affected assets becomes very challenging, complex and time consuming. For this reason, we need to maintain the hierarchical failure and attack propagation structure in a matrix form. In such a hierarchical matrix, the impact of previous hierarchical level is represented on the immediate next hierarchical level (in terms of failure rate/likelihood). The components on the bottom of such trees such as failure causes, or attacks are guiding the design strategies, therefore, we like to see the impact of bottom hierarchical level on all the higher hierarchical level. This can be achieved by normalizing the numerical scores to 1, and then evaluating the higher powers of the matrix until the matrix becomes constant, and then unnormalize the matrix to original scores [33]. A similar matrix can be obtained for attack propagation structure. Both matrices can be combined in bigger matrix for considering cross domain impacts.

# 5. Platform Layer Tools and Methods

In this section we focus on IoT4CPS WP3 contributions for improving dependability on a CPS platform layer. Specifically, we work towards improving system resilience – on one hand we define a method for achieving resilience: Self-Healing by Structural Adaptation. On the other hand, we conduct analysis of indoor direction-finding technologies in order to evaluate its' resilience.

## 5.1 Self-Healing by Structural Adaptation

Failed observation data may compromise system's safety. For instance, an erroneous detection of surrounding vehicles that is input to the path planning unit of an autonomous vehicle might cause fatal consequences. Traditional fault-tolerance aims to overcome such critical failures.

Self-healing adapts the system during runtime to mitigate failures. Adaptation is a challenging process that can be summarized by four abstract steps:

- collect environment information and derive internal system properties (state estimation)
- analyze the observations (failure detection)
- decide how to adapt to reach a desired state (find a recovery strategy, e.g. using an ontology)
- act (recover)

Self-healing is the ability of the system to react also to failures not specifically considered during design-time, e.g., faults caused by functional, environmental or technological changes or zero-day malware. A very promising approach of achieving self-healing is through structural adaptation (SHSA), by replacing a failed component with a substitute component by exploiting implicit redundancy (Fig. 25) [24][25].

The implicit redundancy takes a different approach, compared to explicit redundancy which is achieved by duplicating critical system components and voting over the value of the result. Implicit redundancy is the principle of extracting the missing information about a CPS property from related properties. For instance, the property a, provided by service A, can be substituted by *a'*, which is derived from the combination of properties b and c.

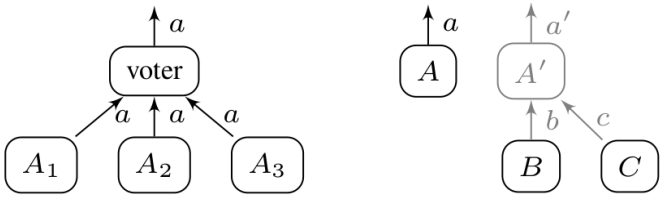


**Figure 25: Types of redundancy - explicit (left) and implicit (right) [25].**

The algorithm uses a knowledge base, modeled as an ontology which defines the interrelation of properties in the CPS as well as additional runtime information of the CPS.

SHSA can be used to replace failed observation data. It monitors and substitutes CPS variables (cf. signals) in messages communicated between application components (e.g., sensors and controllers) based on a knowledge base. SHSA can be encapsulated in one or more components listening and acting on the communication network of the IoT. The detection identifies a failed component by comparing its output to related information on the network using the relations encoded in the knowledge base. A failed component may be removed or shut-down to faulty messages and possibly propagating the failure. Then SHSA spawns a new component – the *substitute* for the failed one. The substitute subscribes to related information and combines these (again by using the relations in the knowledge base) to provide the needed information.

In this project we specifically target extensions to the knowledge base and substitution, the architectural requirements regarding security and the fault detection. Preliminary results include:

- Guided search of a substitution (speed-up of the recovery process). Evaluation and selection of a substitution extracted from the knowledge base [26].
- Architectural requirements for SHSA and considerations w.r.t. security (will be reported in D6.1).
- Extensions to the knowledge base: encode relations in Prolog (rule-based knowledge base to enable requirements checks). Implement state-aware relations (formerly only state-less relations were possible). Demonstration of how to handle time in relations.
- Fault detection: automatic generation of a runtime monitor for related information considering availability of the information and possible time delays (e.g., latency, physics).

Related work is presented in [27] and D2.1: "*Consolidated state-of-the art report*". Case studies and examples can be found in [25], [27] and D6.1.1: "*Architecture for safe and secure automated driving platform demonstrator"*.

### 5.1.1 Architectural Requirements of SHSA

The Architectural Requirements of SHSA are already reported in D3.3. We provide a short reference here, for the sake of completeness.

| # | SHSA Requirement | Rationale |
|---|---|---|
| 1 | Dynamic Composability | Add substitutes, remove/replace faulty components. |
| 1a | Reconfigurable Information Flow | Reconfigurable sender/receiver of messages. |
| 1b | Common Communication | One interface to access information. |
| 1c | Freedom of Interference | Adding a substitute shall not alter the system. |
| 1d | Fault Containment | Avoid fault propagation. |
| 2 | Information Access | For learning, monitoring and substitution. |

**Table 6: Requirements to deploy SHSA on a CPS.**

The system shall be dynamically composable out of several subsystems or components while preventing side effects or undesired emergent behavior. This requirement enables independent development of system components, reusability of components, and reconfigurability of the system during runtime. In adaptive CPS we need dynamic composability in order to be able to add, change or remove components during runtime.

By further refining the dynamic composability requirement we get the following:

- Reconfigurable Communication: The flow of information shall be reconfigurable. Services share information, hence, when a service is added or removed, the communication to and from the service must be installed.
- Common Communication Interface: Since the system is assembled out of several subsystems, a common communication interface must be provided to the reconfiguration mechanism. Hence, SHSA is a challenge especially in heterogeneous systems.
- Freedom of Interference: The reconfiguration mechanism shall not compromise the system's functionality.
- Fault Containment: A failure of a service shall not propagate.

Information Access requirement relates to enabling access to the information related to system state.

### 5.1.2 Extension of SHSA by Context Aware Monitoring (CAM)

This section introduces a new methodology that extends SHSA with the concept of CAM. This methodology allows for resilient control and monitoring. It focuses on increasing the reliability, availability and integrity of collected sensor data by applying SHSA as a self-healing data collection approach in combination with a context-aware data monitoring concept. While Figure 26 shows the proposed concept, detailed information about the methodology can be found in [41]. Hereinafter we focus on the usage of SHSA and skip the concept of context-aware monitoring in this deliverable.

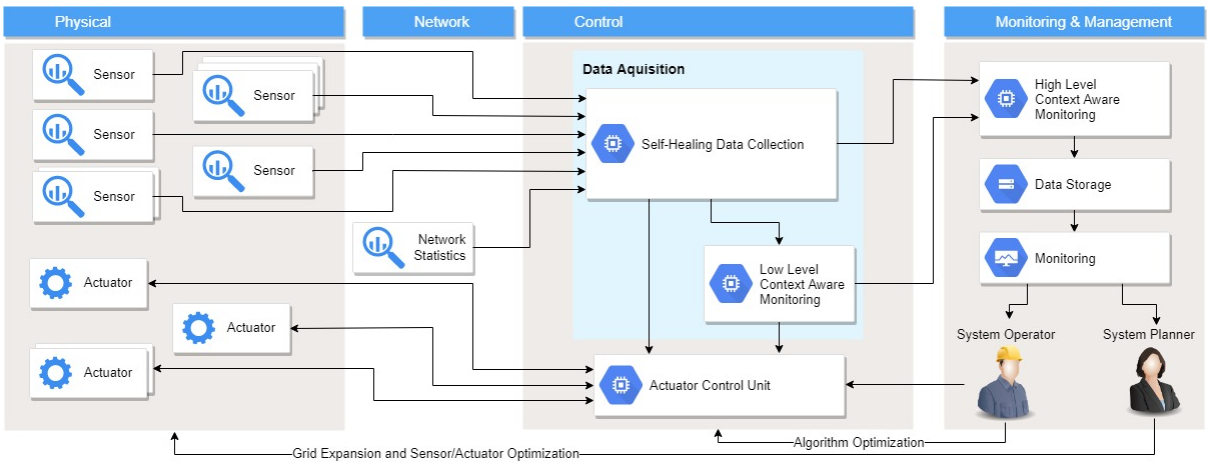


**Figure 26: Resilient methodology for monitoring and control in automation systems (adapted from [41])**

SHSA detects and substitutes faulty observation data (e.g., from sensors, state estimators or other pre-processing units). For instance, consider the amperemeter measuring the current through a power line fails, then an overload of this line cannot be detected by a monitor relying on the data of this amperemeter only. SHSA compares related signals to detect failures (here, in the amperemeter messages), and substitutes failed signals by a function of related signals, in order to provide reliable observation data to subsequent units (here, overload monitor). To this end, SHSA maintains a knowledgebase of information redundancy. Figure 27 depicts a possible SHSA knowledge base for the respective application domain. The knowledge base comprises the relations between CPS variables $v$ and the availability of corresponding signals $v$ (observations over time).

The CPS variable $v_p$ represents the power consumption on the power line $l_1$. The power consumption is measured by a sensor providing the signal $v_p$. The same signal can also be derived from $v_u$ and $v_{i1}$ (via relation·, cf. implicit redundancy). Signals $v_{i1}$ and $v_{i2}$ measure the same CPS variable $v_i$ (cf. explicit redundancy). For instance, an energy meter at the consumer and an amperemeter of the provider at the house connector both measure the current. Variables can be provided by constants or configuration parameters too (e.g., voltage $v_u$ or modes $v_{switches\_status}$). SHSA is implemented as an additional service acting on the communication network of a system (monitoring messages and substituting signals in messages). The SHSA service may run on a separate platform and connect to existing systems, when the requirements of dynamic composition and communication are fulfilled. The knowledge base is adaptive, that is, the relations and availability of signals may change during runtime, in order to cope with various system changes.

This example shows how the new approach can be applied to energy distribution systems, but it can similarly be applied to other application fields.

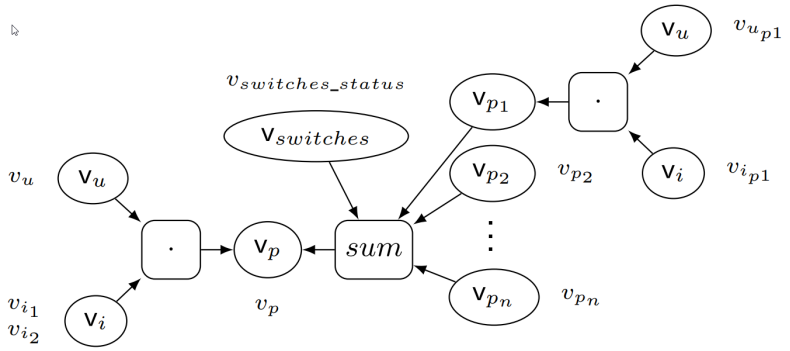


**Figure 27: Example SHSA knowledge base for the smart grid. CPS variables (ellipses) are related via functions (boxes) to each other. Variables are annotated with available signals *v*.**

## 5.2 Resilience analysis of indoor direction-finding technologies

As outlined in deliverable 3.3 a wide range of system critical applications in various domains such as autonomous driving or industry 4.0 actively utilize localization. To apply localization mechanisms in the context of these applications, it is important to assure their secure and dependable performance. Depending on the application requirements (complexity of the environment, accuracy, energy footprint, update rate, etc), different localization technologies can be utilized. As discussed in deliverable 3.3, several potential attack vectors exist in the context of localization technologies. It is therefore important to evaluate a technology and potential counter measures against existing threats within the designated application before its setup.

In Deliverable 3.4, a specific localization approach relying on distance estimations (with a single anchor), was presented, which can be hardened against attacks. Such approaches became popular due to the increased accuracy achieved by the state-of-the-art technologies estimating distance, such as Ultra-wideband (UWB). The recent trend of estimating the Direction of Arrival (DoA) of an incoming signal, which may facilitate the setup of Real-Time Localization Systems and decrease their costs, may also face security challenges, as already pointed in [42]. Prior to investigating the security aspects of technologies enabling DoA, it is crucial to understand their features and limitations.

As deciding on a specific localization technology is highly context dependent, the process of selecting such a technology should be guided by certain design decisions. When enough information of certain technology is available this can be done using established knowledge on the technologies. However, given the rapid development in the context of wireless communication and its various applications in the context of localization, this information is often not available. This deliverable presents the steps in designing an experimental setup, which can be used to investigate the suitability of direction estimation mechanisms. Furthermore, the setup will allow to benchmark and compare different (competing) technologies. The methodology will be outlined in the context of two localization approaches out of the Direction of Arrival (DoA) localization class, namely UWB and Bluetooth Low Energy (BLE). However, the methodology can be easily extended to test and benchmark other DoA based technologies, such as WiFi.

This contribution focuses on pinpointing important factors in the preparation of localization test / benchmarking test setups. Therefore, the results are only briefly explained. However, a detailed discussion and analysis of the results can be found in deliverable D5.4.2, where an implementation of the proposed experiments is used to establish how suitable UWB localization is in the context of enriching indoor context awareness.

### 5.2.1 Introduction to Direction Finding

Direction finding is useful for many applications such as estimating the direction of an emitter relative to a specified direction [43]. Technologies for direction finding typically make use of antenna arrays and direction-finding methods such as Beamforming and Multi-Signal Classification (MUSIC) [44].

Currently, the applicability of direction finding to state-of-the-art communication technologies has attracted the attention of the industrial and the scientific community. This specially holds for WiFi, Bluetooth Low Energy (BLE) and Ultra-wideband (UWB) technologies.

These technologies are currently used in Real Time Localization Systems (RTLS), mostly based on distance estimations from multiple reference devices (typically called anchors). Supplying an additional direction of arrival (DoA) information could enhance such systems by reducing the number of anchors, reducing the energy consumption, and increasing accuracy and precision. Currently UWB is typically used for high accuracy indoor applications, as it achieves sub-decimetre accuracy (Großwindhager, et al., 2018). In contrast BLE typically has a localization error in the order of a few meters. To select and apply any of these technologies in the context of trustworthy localization applications is necessary to understand their pros and cons.

So far angle of arrival (AoA) and phase difference of arrival (PDoA) transceivers had limited applicability, due to the inherent complexity of such modules and their commercial prices. This scenario is changing as there are currently off-the-shelf devices providing PDoA-based angle estimation for BLE and UWB transceivers. Also, a new standard has been developed for BLE which supports direction finding: BLE 5.1. We refer to angle of arrival (AoA) as a method enabling the estimation of the direction of an incoming signal, which can be done, for instance, by using directional antennas. By phase difference of arrival (PDoA), we refer to a specific method which can be used for AoA estimation in which the phase difference between two signals are used.

Although the benefits provided by direction finding to these technologies are evident, so far these technologies have only been subject to initial evaluations within the scientific community. In addition, no thorough comparison/benchmarking has been conducted. This contribution aims to design experiments which allow for a fair comparison of the AoA capabilities of BLE and UWB with a focus on indoors applications. It is intended to guide researchers and developers through important aspects that should be considered when evaluating localization technologies. Based on the experimental setup established in this deliverable, D5.4.2 presents some preliminary results about the evaluation of the UWB module obtained with the methodology described here.

### 5.2.2 Methods for angle-of-arrival estimation

Although both UWB and BLE rely on the phase difference between two signals to estimate the incoming angle of a RF signal, they use different approaches. In this subsection, we summarize the main aspects necessary to understand the basic principle behind each technology. We limit our introductory explanation to these two technologies, as it would be out of scope to survey all the existing DoA methods.

UWB

We aim to evaluate an approach which is described in details in (Dotlic, Connell, Ma, Clancy, & McLaughlin, 2017), because (to the best of our knowledge) it is the only documented module, which features UWB-based PDoA estimation. To date there is no standard encompassing direction estimation within UWB. Typically, an UWB receiver provides a Channel Impulse Response (CIR), estimated from the preamble of an incoming UWB packet, as shown in Figure 28. By using coherent receivers, both absolute value (as shown in Figure 28) and phase information can be extracted. In particular, the First Path (FP) sample phase, which corresponds to the phase in which the straight line connecting transmitter and receiver, can be estimated. Then, the construction of the PDoA

UWB receiver is nothing else than two instances of such a receiver placed side-by-side at a given distance (which should be less than half of the wavelength used) supplied by the same clock source. Both receivers turn on at the same time and calculate the FP phase independently, and the times of arrival of the signal in both receivers are subsequently subtracted from one another. Finally, the AoA can be calculated by:

$$\theta = \arcsin\left(\frac{\lambda\,\alpha}{2\,\pi\,d}\right)$$

Where:

$\theta$ = angle-of-arrival

$\lambda$ = wavelength of the UWB carrier

α = phase-difference between the FP estimated by both receivers

d = distance between the two antennas


**Figure 28: Exemplary Channel Impulse Response (CIR) obtained from a UWB receiver.**

A module implementing this approach, namely dwm1002, was distributed by Decawave and used in our experiments. Additional details can be found in (Decawave, Beta PDoA kit User Manual V1.0., 2018).

Bluetooth Low Energy

The recent BLE 5.1 standard (Core specification v5.1, 2019) describes the optional feature of supporting AoA/AoD (angle-of-departure). This can be accomplished by appending a constant tone extension (CTE) to the end of a packet. The constant tone is sampled by a distant coherent receiver using multiple spaced antennas (connected to the BLE transceiver through a switch). Therefore, the samples are not acquired at the same time instant (as is the case for UWB), but sequentially, which makes the approach more vulnerable to clock variations. A higher accuracy in angle estimation could be achieved if two BLE receivers were used in parallel in the same module. Such an approach has not yet been evaluated (up to our knowledge), but is less likely to be accepted, as BLE is typically intended for (low-power) communication in the first place and using multiple antennas would lead to an increase of costs and energy budget. Additional details such as how many antennas should be used, or in which sequence the samples should be taken are out of the scope of the standard and left as a degree-of-freedom to the developer. To our knowledge, the first off-the-shelf development kit supporting AoA estimation with BLE 5.1 (and the only one available at the time of the start of this investigation) was the SimpleLink™ Angle of Arrival BoosterPack (Texas Instruments, Bluetooth™ Angle of Arrival (AoA) Antenna Design) developed by Texas Instruments. We use this module in our evaluation.

### 5.2.3 Security threats

To the best of our knowledge, there are no specific published attacks applicable to direction of arrival estimation using UWB. However, there are general attacks to Angle of arrival estimation, such as Jamming (A. Abdelaziz, 2016), which may be also applicable to UWB. This is due to the novelty of the approach. In addition, it is difficult

to perform an attack, as an adversary must be able to change the phase difference of the transmitted signal measured at two antennas quasi-simultaneously. Such an attack is easier to implement in the context of BLE, where an incoming signal reaches the diverse receiving antennas with a significant time delay.

In order to compromise an angular estimation by a BLE compliant receiver, an adversary transmitter can control its CTE phase [42], which should be always constant according to the standard. Additionally, to this attack, which we may reference as a *Direction Fraud,* it is possible to think in a *Mafia (direction) Attack,* which consists of an external device interfering in the angle estimation from the anchor towards the tag. We will discuss how to design such attacks in Subsection 5.2.5 .

### 5.2.4 Goals

We aim to design and experimental benchmarking setup, that enables us to:

- To evaluate the performance of AoA-capable technologies in realistic environments. Independent evaluations investigated the two target technologies, including the one published by the manufacturers, and showed how the modules behave in a perfect environment (anechoic chamber), where the absence of multipath/reflections and obstacles may overestimate the performance of the technologies in real-world applications. These evaluations are often insufficient to draw strong conclusions on which technology to choose for a particular application. Therefore, we design measurement setups intending to provide developers, researchers and system designers aiming to perform their own comparison across different DoA technologies with guidelines and recommendations on how to perform a fair comparison across DoA technologies and recommendations;
- To compare the DoA resilience of different technologies in non-line-of-sight (NLOS) and realistic multipath setups. The influence of NLOS and multipath are often jointly evaluated in complete localization systems only. Nonetheless, the decoupling between distance and direction estimations enables to evaluate their performance separately. BLE-based distance estimation is well-known to be strongly affected by multipath effects (Zanella & Bardella, 2014). UWB, on the other hand, is capable of distinguishing quasi-simultaneous pulses due to its narrow pulses (which are a consequence of the high bandwidth) in the time domain. Additionally, the high bandwidth of UWB makes it more resilient to frequency-selective obstacles than BLE. By combining strong multipath with NLOS in the same experiments we intend to evaluate the feasibility of estimating the incoming angle from the multipath components, as the First Path component will be totally blocked (under hard NLOS). The same setup still provides significant results to other technologies.
- To test other hypotheses that are relevant in the domain. In particular, about the possibility to improve accuracy by varying parameters and about the potential benefit of using multiple antennas to increase the accuracy of angle estimations.

Though the results of the experiments will not be presented in this deliverable, partial results will be made available in the Deliverable 5.4.2 of IoT4CPS, where we used the testbed setup developed in WP3 to benchmark the DoA performance. The complete set of conclusions and relevant results will be submitted for publication in a recognized dissemination outlet.

### 5.2.5 Experiments

We describe in this Subsection the complete set of experiments designed to evaluate and compare DoA technologies. To test them in their natural habitat it is important to utilize a parameterization as recommended by the manufacturers. For UWB the standard parameterization is as follows:

- Preamble length of 128 symbols
- SFD of 8 symbols
- PHR of 16 symbols
- No data in the payload
- Data rate equals 6.8 MHz
- Channel 5 is used
- 2 antennas only

And for BLE, the standard parameterization is as follows:

- Connection interval equals 50 ms
- All channels in the range 0-37 were enabled
- CTE length equals 160 us
- Sampling slots of 1 us are used
- 3 antennas (a comparison with 2 antennas is also included)

To evaluate solely the impact of the multipath components, we designed experiments in different indoor environments. In particular, we consider a classroom, which is expected to present discrete multipath components due to multiple spaced objects, and a corridor, which is expected to present a continuous range of multipath components. The resilience of each technology to NLOS is then evaluated in both of these environments. We also design experiments to evaluate the effect of varying parameters intrinsic to each approach, such as the number of antenna elements and the signal bandwidth.

Line-of-sight

In order to compare the performance of technologies in realistic environments, we started by designing an experiment taking place in a common environment, sharing spectrum with other technologies (in this case, WiFi), which is a classroom. This environment is occupied by several distinct objects, such as tables and chairs, which are also commonly found in offices and restaurants.

By moving the transmitter around a fixed receiver/anchor, it is possible to register a difference in phase among elements of the antenna array. The same effect can be generated by rotating the anchor in the opposite direction while fixing the transmitter in a static position. Not only is this approach more practical, as depending on the environment, it is the only feasible option due to spatial constraints and is often used in anechoic chambers. We recommend using the same approach, as it allows a seamless comparison with results obtained in an anechoic chamber, as well as with results from other experiments in which it is not possible to rotate the tag.

As the current COTS hardware supporting PDoA feature highly directional antennas with a non-ideal performance near the fire-ends (the axis of the antenna array) an angular range from −90° to 90° in steps of 10° should be sufficient for this kind of comparison. As the technologies develop, this step size may no longer suffice. Additionally, we encourage the repetition of the measurements at different sender-receiver distances due to the high complexity of the environment. Distances in the range, 2m, 3m and 4m should be common in such a scenario. A photography illustrating our setup for this design can be seen in Figure 29 and an extract from the results is shown in Figure 30. The complete set of results obtained with UWB is available on Deliverable 5.4.2.

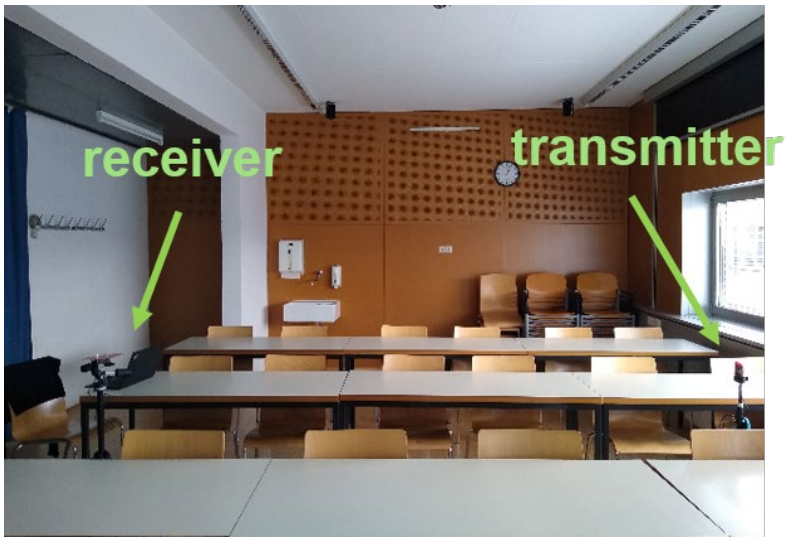


**Figure 29: Setup of measurement under LOS in a classroom when the distance between sender and receiver equals 4m and the angle of the Rx (anchor) equals 0°, i.e., the antenna array is perpendicular to the line connecting sender and receiver.**

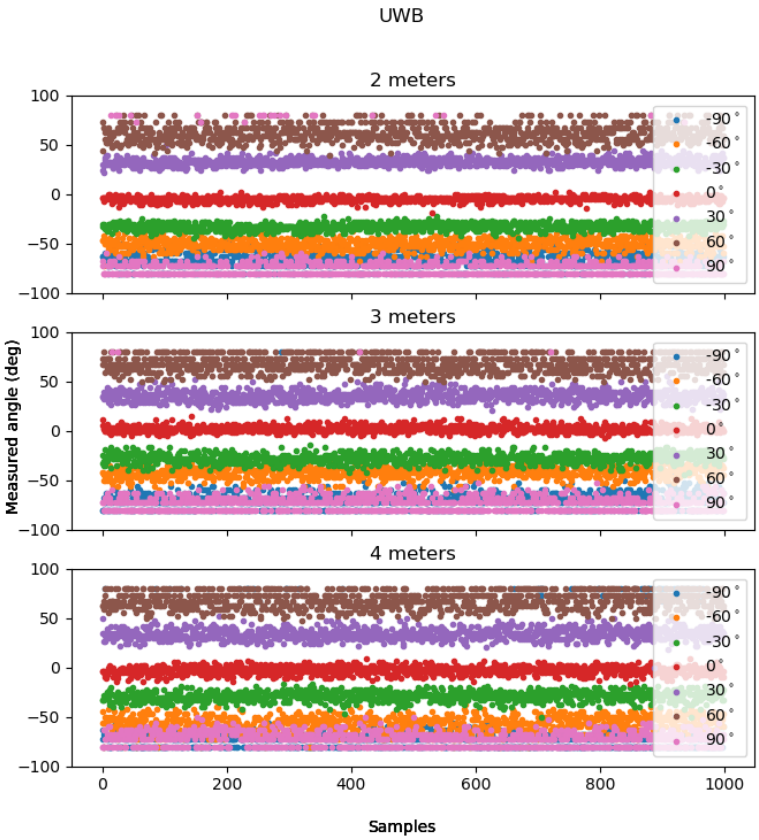


**Figure 30: Angles estimated with UWB at 3 different distances in a classroom.**

Non-line-of-sight

In order to assess the effect of removing partially and totally the first path component we propose inserting different obstacles in between sender and receiver. A perfect absorber/absorber foam (worst-case scenario,

which we also refer to as strong NLOS) and a human (common obstacle in diverse applications, which we also refer to as light NLOS) may be chosen as obstacles. A single distance may be fixed in order to reduce the number of parameters to analyse. We opted for a sender-receiver distance of 3m. Additionally, the angular range and step size may be modified accordingly for the same reason. We recommend measuring at 3 different angles with symmetry to facilitate debugging. In our experiments, we used: −30°, 0° and 30°. The obstacle is positioned always in the middle between sender and receiver, i.e., at a distance of 1.5 m of any of the devices.

Strong multipath

To assess the performance of angle estimations subject to multipath components, it is important to diversify the environment. As already mentioned in Subsection 5.2.5, an environment may introduce a continuous and uniform range of multipath components in the signal reaching the receiver device. Such an environment can be emulated by a long and narrow corridor, as the one shown in Figure 31. In our measurements, the distance between tag and anchor was increased to 10 m. Although this prevents a seamless comparison with the results from the previous Subsections, it decreases the time difference between multipath components and the first path, therefore amplifying the expected effect: interference between first path and multipath. We test again for 3 different angles between anchor and tag as in Subsection 0: −30°, 0° and 30°.

Multipath and NLOS

Similarly to the “Non-line-of-site” design, this setup is intended to eliminate (partially or totally) the first path component. Once again, two obstacles with different properties may be placed in between sender and receiver. Besides comparing the performance of the technologies, this test assesses the capabilities of technologies to estimate the incoming angle form multipath components. A photography of the setup is illustrated in Figure 31.

**Figure 31: Rear view of the (UWB) tag. Perfect absorber is placed at the same height in such a way that the first path is obstructed.**

Number of antenna elements

It is known that by increasing the number of antennas it is possible to get more accurate direction estimations, for instance, by using the MUSIC algorithm [44]. However, in [45] inconclusive experimental results were found by increasing the number of antenna elements. This finding motivates us to design an experiment in which it is

possible to test whether adding spatial diversity increases accuracy and/or precision of angle estimations. Due to hardware constraints, the experiment has to be limited to BLE, as the available module provides means to easily check this hypothesis. We recommend performing the angle estimations outdoors or in a multipath-free environment, to avoid multipath components.

Variation of parameters

Besides the parameters varied in the previous designs, several other intrinsic and technology-dependent parameters can interfere in the performance of angle estimations. As the target technologies within this study and their respective methods to estimate angles are very different, we group the key parameters per technology. For BLE, they are: CTE length, sampling slot duration, sampling frequency and number of channels. For UWB, these include preamble length and channel (with different bandwidths, but same central frequency)

Varying channels/central frequency within UWB is not included as it requires additional packets and is potentially affected by mutual coupling between the antennas, although it may become a hot research topic in the future, with the development of application specific antennas. Note that the channel variation within UWB can be tested as long as the central frequency remains the same.

Security

In order to test security aspects, a software-defined radio (SDR) is recommended, as it is more flexible than COTS devices and typically enable the implementation of more sophisticated attacks, which would not be possible using standard-compliant modules. This approach was also chosen in [42]. Thus, a BLE Direction Fraud can be accomplished by using a SDR at the transmitter's side instead of the COTS devices used in our setup. Nonetheless, if the adversary intends to accurately control its angle towards the receivers, it must know not only the switching sequence at the receiver, as well as its actual angle towards this device. Cominelli et al [42] show a counter-measure to this attack.

The BLE *Mafia (direction) Attack* can be implemented by adding to the setup from our experiments described in Subsection 5.2.5 an adversary aiming to change the apparent direction of the honest transmitter. It can accomplish this by synchronizing to a packet transmitted by the sender and either overshadow its CTE with a modified and stronger one or adding to the transmitter's CTE another one which when combined results in a carrier with the desired phase. Up to our knowledge, none of those (Mafia) attacks have been implemented to date. Jamming attacks affecting either of the technologies can also be possibly implemented by using SDRs. It is important to account for the bandwidth and frequencies covered by a SDR (or its daughterboard) so that it fits to the experiments. The costs of SDRs tend to increase with these parameters, which makes SDRs covering the UWB spectrum expensive.

## 5.3 Estimating Orientation

In this subsection, we discuss the challenges in estimating orientation, based purely on inertial sensing devices, and provide experimental results recorded during operation of a realistic work tool prototype in harsh industrial environment. The measurement noise, inherent in various commercial inertial sensors, is typically modelled with a Gaussian distribution and usually, for smoothing the orientation estimates, a Gauss-Newton optimization is applied. In harsh environments, it is critical to obtain not only the orientation estimates, but also to estimate their uncertainty. In the case of Gaussian noise, the uncertainty is characterised by the covariance. Typical optimization algorithms for smoothing orientation estimates require an initial estimate, inertial data from accelerometers ($y_a$), gyroscopes ($y_\omega$) and magnetometers ($y_m$), and the corresponding covariance matrices ($\Sigma_a$, $\Sigma_\omega$, $\Sigma_m$). In the filtering process, the covariance of the estimate plays a critical role for computational reasons,

for which specific diagonal subsets of the inverse are considered. Specific specialized filtering methods, such as the extended Kalman filter (EKF), can be interpreted as Gauss-Newton optimization of the filtering problem using a single iteration with a step length of one. Alternatives to EKFs for orientation estimation are complementary filters, which benefit from the fact that both the gyroscope and the combination of the accelerometer and the magnetometer provide information about orientation. The orientation estimates obtained from accelerometer and magnetometer data are noisy, but accurate over time. On the other hand, the orientation estimates based on gyroscope data are accurate on a short time scale, however they drift over time. These properties can be interpreted and exploited in the frequency domain. The orientation estimates using gyroscope data have desirable properties at high frequencies and hence are filtered using a high-pass filter. In contrast, the orientation estimates obtained from accelerometer and magnetometer measurements have desirable properties at low frequencies and hence are filtered using a low-pass filter.

The purpose of combining orientation and localization awareness is to enable specific corrective measures and/or provide predictive feedback related to position, location, distance and angle orientation of the work tool and the hand with respect to each other and with respect to the working piece. In this scope, we explored the accuracy and resilience of orientation estimates for a particular industrial tool, i.e. a power tool, in the context of a realistic harsh industrial environment.

For this purpose, we have mounted one wireless (matchbox-sized) Xsens IMU device (Figure 32/left) on top of a commercial power tool (Figure 32/right) and performed various experiments with the augmented tool in harsh environments, including large disturbances of high-frequency vibration and electromagnetic interference. The time series fragments in Figure 33 present a sequence of short drilling operations followed by brief rotations of the work tool around its axes. Both raw sensor data and orientation estimates are computed and displayed in real time. As this live experiment did not include any additional source of information (e.g., camera), there was no ground truth orientation reference for computing the error of the estimate.


**Figure 32 (left) Xsens MTw IMU, providing wireless motion tracking by built-in 3-axis accelerometer, gyroscope and magnetometer. (right) Power tool augmented with an Xsens on top.**

Nevertheless, as Figure 33 suggests, despite the large magnitude of disturbances induced by the motor rotation, on overall the deviation of the orientation estimate varied between 1 and 2 degrees for pitch and roll, and was below 5 degrees for yaw, providing evidence for a reliable and stable orientation estimate in industrial operation.

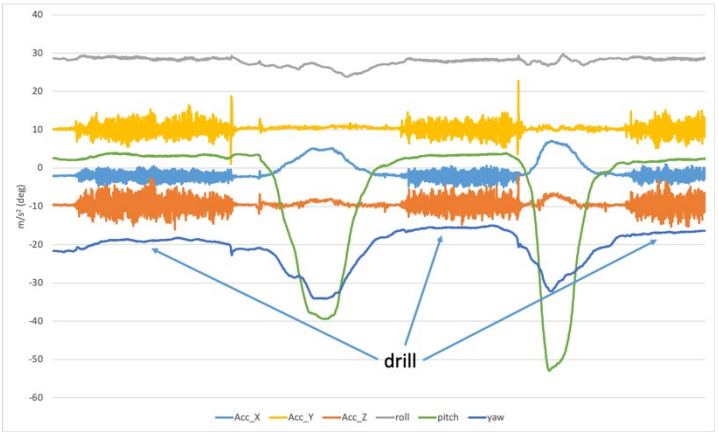


**Figure 33 Acceleration X,Y and Z axes (in m/s$^2$) and orientation estimate roll, pitch and yaw angles (in deg) time series, in a drilling experiment using Xsens IMU attached on top of a power tool. The drilling fragments demonstrate the robust orientation estimate in harsh industrial conditions.**

The overall distributions of the time series, presented in Figure 34, are in line with the fact that it is typically easier to obtain accurate roll and pitch estimates than it is to obtain accurate heading (yaw) estimates. Although the accelerometer and the magnetometer measurement noises are of equal magnitude, the heading angle is estimated with less accuracy compared to the roll and pitch angles. The reason for this is twofold. On one hand, the magnetometer's signal to noise ratio is worse than the one for the accelerometer, since magnetometers have a magnitude of 1 while accelerometers have a magnitude of 9.82 m/s$^2$. Furthermore, only the horizontal component of the magnetic field vector provides heading information. Since accelerometers only provide inclination, in case no magnetometer measurements are available, the heading can only be estimated from gyroscope data, which inevitably drifts over time. The presented results suggest that depending on the magnitude of mechanic and electro-magnetic disturbances and the uncertainty in inertial and magnetometer measurements, the errors in roll and pitch (and yaw to a certain degree) estimates, therefore, can be kept under control effectively.

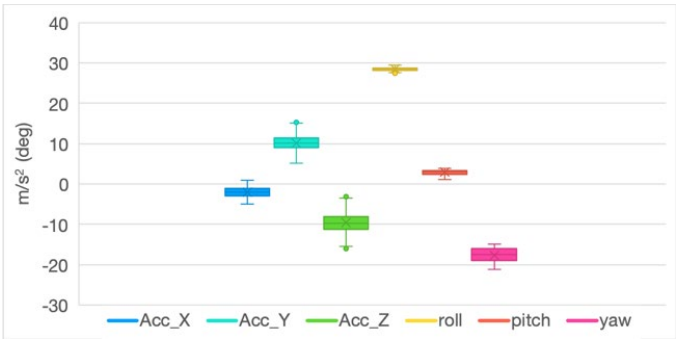


**Figure 34 Overall distribution of the time series presented in Figure 33, revealing the highly accurate roll and pitch estimates and the less accurate heading (yaw) estimates (in deg).**

# 6. Network Layer Tools and Methods

In this section we focus on IoT4CPS WP3 contributions for improving dependability on network CPS layer. Specifically, we focus on a recommender system which should help the IoT system architects to select the protocols which meet the requirements of their systems.

## 6.1 Recommender System for Dependable IoT applications

Along with the growing market of Industrial IoT (IIoT) applications, the set of available network technologies is continuously expanding. Today, developers have a huge set of connectivity networks at their disposal, ranging from short-range networks such as Bluetooth to global connectivity via satellite networks [1]. Depending on the specific use case of each IIoT application, different approaches constitute the most cost-effective network technology solution, as there is no “one-size-fits-all” solution.

Choosing the set of network technologies, which fits the needs of the IIoT use case must be a careful trade-off between the ability of the technologies to meet specific functional requirements and the related costs [1]. Complex IIoT systems — as sketched in Figure 35 – have a significantly larger set of dependability requirements compared to “normal” IoT applications. As the systems connectivity network plays a major role in fulfilling these requirements, choosing the correct set of technologies is crucial.

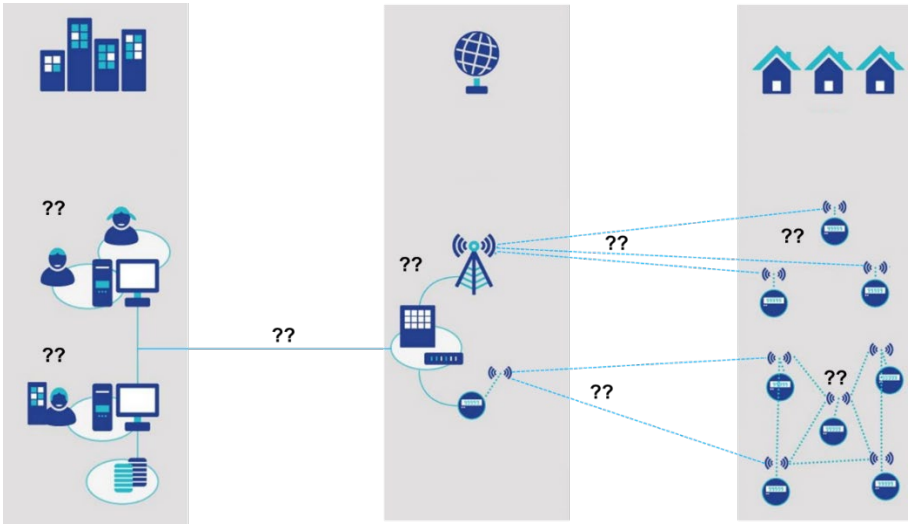


**Figure 35: A complex IIoT system comprising file, edge and cloud components**

## 6.2 Aim of the Recommender System

Depending on the application and its scope, different system architectures can be applied. However, most applications follow the generic system architecture. This architecture comprises different system levels and possible interconnections.

The challenge in building such systems lies on interconnecting already ongoing engineering activities and brownfield devices at multiple levels and enrich them with tools to address dependability aspects of the system. Usually such a system design is derived by experts and causes a lot of effort and state-of-the-art expertise. The proposed approach tries to perform the design and partial configuration of a system in a semi-automatic way. Thereby, specific system constraints and requirements will be considered. They are ranging from basic

communication properties such as energy consumption, bandwidth and latency to specific dependability attributes such as integrity and availability.

The system designer has to provide application specific requirements (e.g., purpose, location, connectivity, power supply) and high-level architecture patterns (e.g., direct connection field cloud or multilevel communication via edge devices). Based on this information, the recommender system is capable of computing a feasible system setup (system topology, technologies to apply within the system) regarding a specified use case, without the need of consulting experts. Figure 36 illustrates the workflow of the recommender system.

Beside generating a suggested system topology, the recommender system will match the application's demand regarding dependability attributes with the offered attributes of the possible technologies. Thereby, qualitative and quantitative measures are applied. Based on this, a final selection of technologies is possible.

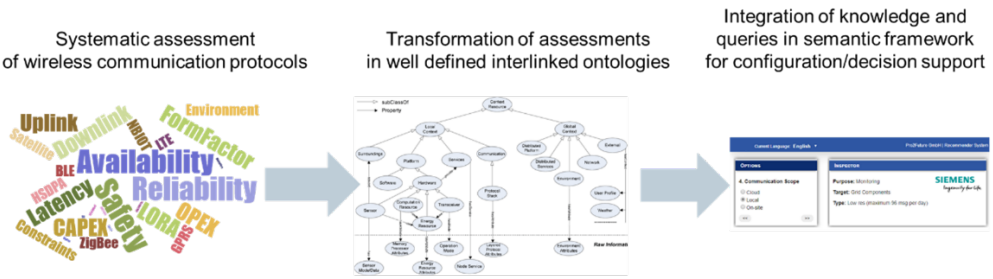


**Figure 36: Illustration of the workflow of the recommender system**

## 6.3 Knowledge Base

The aimed ability of the recommender system is based on maintaining a knowledge base, comprising of the technical and functional characteristics of all the available network technologies. The database of the recommender system is based on the Open Semantic Framework (OSF). The open semantic framework, cf. [34], structures information in domain-specific knowledge packs (KP), which depend on several core ontologies (containing general information about concepts, quantities, units, events etc.). The OSF allows for a modular management of knowledge specifically tailored to the application as sketch in Figure 37. OSF knowledge can be accessed and managed during runtime via a REST API and thus easily integrated in established process environments. Using construct or update features, it is possible to curate and extend both the available knowledge in the OSF as well as construct or update the corresponding queries. Additional domain-specific knowledge packs and domain specific queries can be added to extend the functionality of the framework towards the desired application domain.

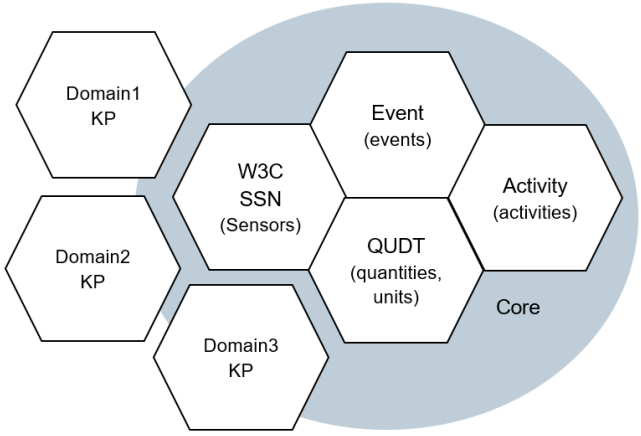


**Figure 37: Modular architecture of the Open Semantic Framework**

Based on the OSF different KP were developed as basis for the recommender system as depicted in Figure 38. The "protocol KP" comprises the basic properties of communication technologies. Besides that, the KP focusses on measures regarding dependability. These measures contain a set of dependability methods such as multipath routing, frequency hopping, or retransmit mechanisms. In addition to the method, also the layer (w.r.t. the ISO-OSI reference model) is specified. The dependability measure of the protocol KP is further linked to concerned threats of the dependability KP. Threats are basically described in terms of affecting dependability attributes and their impact on the overall system. The impact in this KP is agnostic w.r.t. to application domains or systems.

If a more complex modelling is necessary, a KP based on the threat model introduced in Section 4 and 5 can be designed. With this model a higher level of detail can be achieved than in the basic model. The OSF enables to curate dependability threats independently from protocols (different knowledge packs). Additionally, the links across different knowledge packs are used to assess how well specific protocol instances can deal with threats. This allows an assessment of the overall dependability of a set of protocols across different threat levels on different layers.

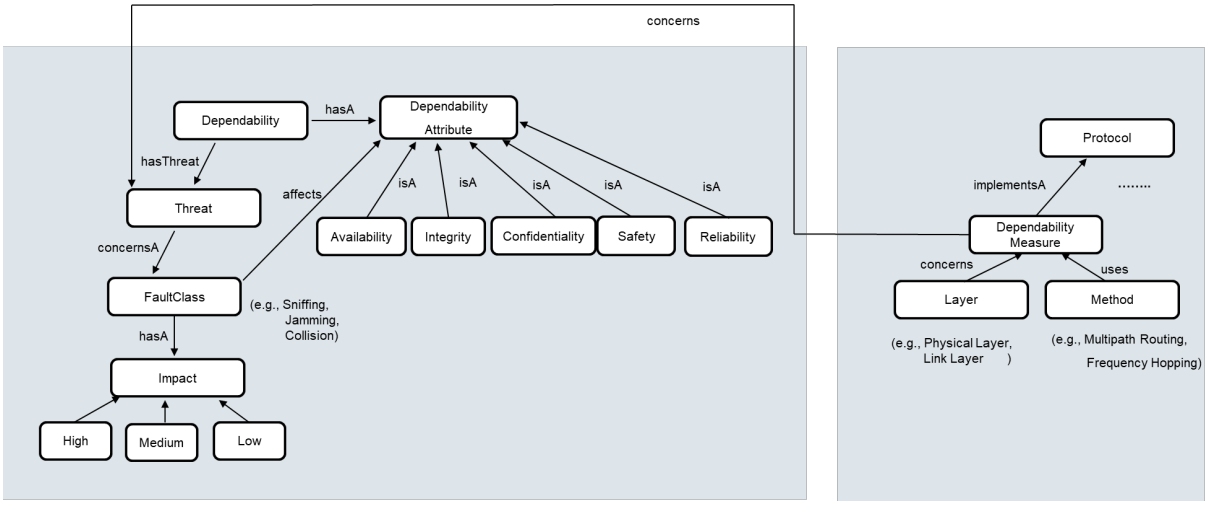


**Figure 38: Core KP for the recommender system**

## 6.4 User Interface

In order to sketch the functionality of the recommender system, an example a user interface is depicted in Figure 39. On the left-hand side, it allows the system designer to select the environment high-level requirements. According to this selection, the centre of the UI reflects the architecture of the overall system. Additionally, it is

possible to set specific numerical parameters for the overall system on the right-hand side. The bar on the top finally depicts considered. According to the traffic-light colouring, the technologies are justified.

Although a stand-alone solution with a dedicated user interface suffices the needs of typical systems, more complex systems can hardly be modelled. However, the limited flexibility and complexity is mainly limited by the expressiveness of the user interface. Additionally, commonly used system design tools are easier to manage for a system designer. Thus, the integration of the business logic of the recommender system in state-of-the-art system design tools is investigated.

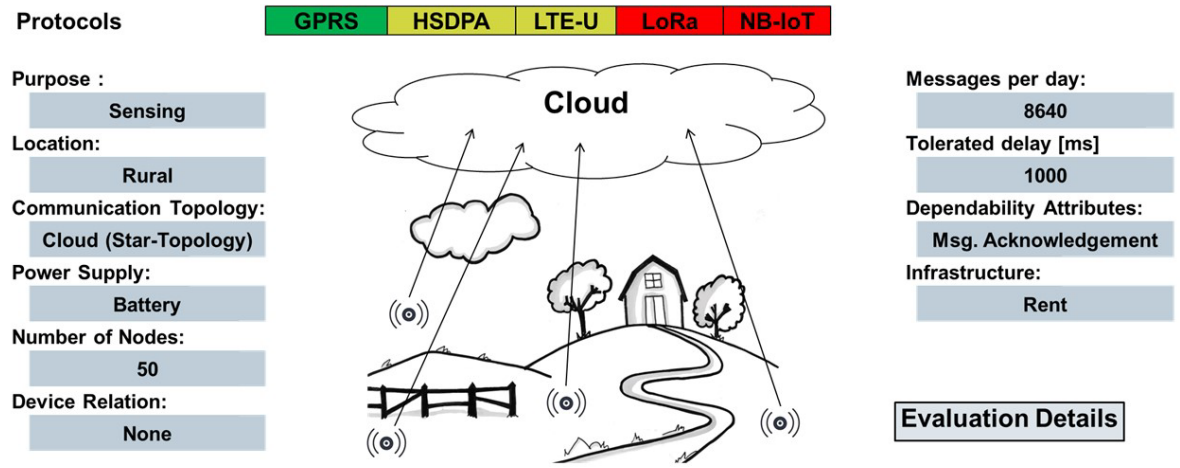


**Figure 39: Illustration of the Recommender System with possible input parameters**

### 6.4.1 Integration in the IOT4CPS framework

As a next step, the recommender system and the respective UI concept should be integrated in a state-of-the art system design tool. Thereby, the used set of tools should comply with the design framework developed in the context of this project. Like the development of "THREATGET" [1], the recommender system can be extended or connected to a plugin for the well-known tool Enterprise Architect. This tool allows for flexible system design and a seamless use of different components of the IOT4CPS design framework in a single environment. The environment itself is extensible such that the business logic of the recommender system can be integrated. Additionally, many domain specific frameworks are based on Enterprise Architect as well which allows for a comprehensive and seamless system design. For example, in the Smart Grid domain the toolset for the Smart Grid Reference Architecture (SGAM) is based on Enterprise architect [2].

### 6.4.2 From development to operation

Traditional automation and control systems strictly distinguished between design/development on the one hand side and operation on the other hand side. Due to the increasing speed of technology development in the IOT, this border vanishes successively even in the context of highly reliable and safety critical systems. Although "DevOps" concepts are becoming more and more relevant, dependability aspects have to be guaranteed at any time of operation.

[1] https://www.threatget.com/what-is/

[2] https://sgam-toolbox.org/

In order to address this aspect, the recommender system can evolve from a pure design system to a DevOps system comprising run-time dependability evaluation and system adjustment in case of dynamic system changes or faults. For this purpose, methods for monitoring and assessing system parameters need to be developed. These methods serve as input to the recommender system are the basis for runtime reconfiguration of the system.

# 7. Conclusion

Deliverable "*D3.7: Final report on tools and methods"* provides a detailed insight into state-of-the art tools and methods, which are the outcome of IoT4CPS project WP3. The methods and tools are strongly motivated by our two main use cases of Automated Driving and Smart Production. The aim of methods and tools of D3.7 is to build dependable systems by improving their safety & security using:

- Security tools such as ThreatGet, GSFlow, Moreto
- Security risk assessment methods such as FMVEA
- Co-engineering methods for safety and security
- Resilience techniques and architectures, such as Self-Healing by Structural adaptation
- Trusted localization and orientation
- Recommender Systems for building large IoT-based applications
- Sensor-level security
- High-level guidelines on writing crypto APIs
- V&V patterns for Security Risk Assessment

Our tools and methods cover a wide spectrum of both design time and runtime methods which aim to make the system more safe, secure, resilient to failures, reliable and trustworthy in general. We strongly believe that dependability is a complex system property which must be addressed at different levels of complexity and multiple stages of product lifecycle. IoT4CPS consortium is looking forward to further improving the tools and methods in order to provide the industrial partners in Austria the means to achieve competitive advantage on the market.

# 8. Appendix A: V&V pattern – Security Risk Assessment with Attack Trees

In this appendix we provide a formalized abstraction, namely a V&V pattern, for our Security Risk Assessment from section 7.1.1.

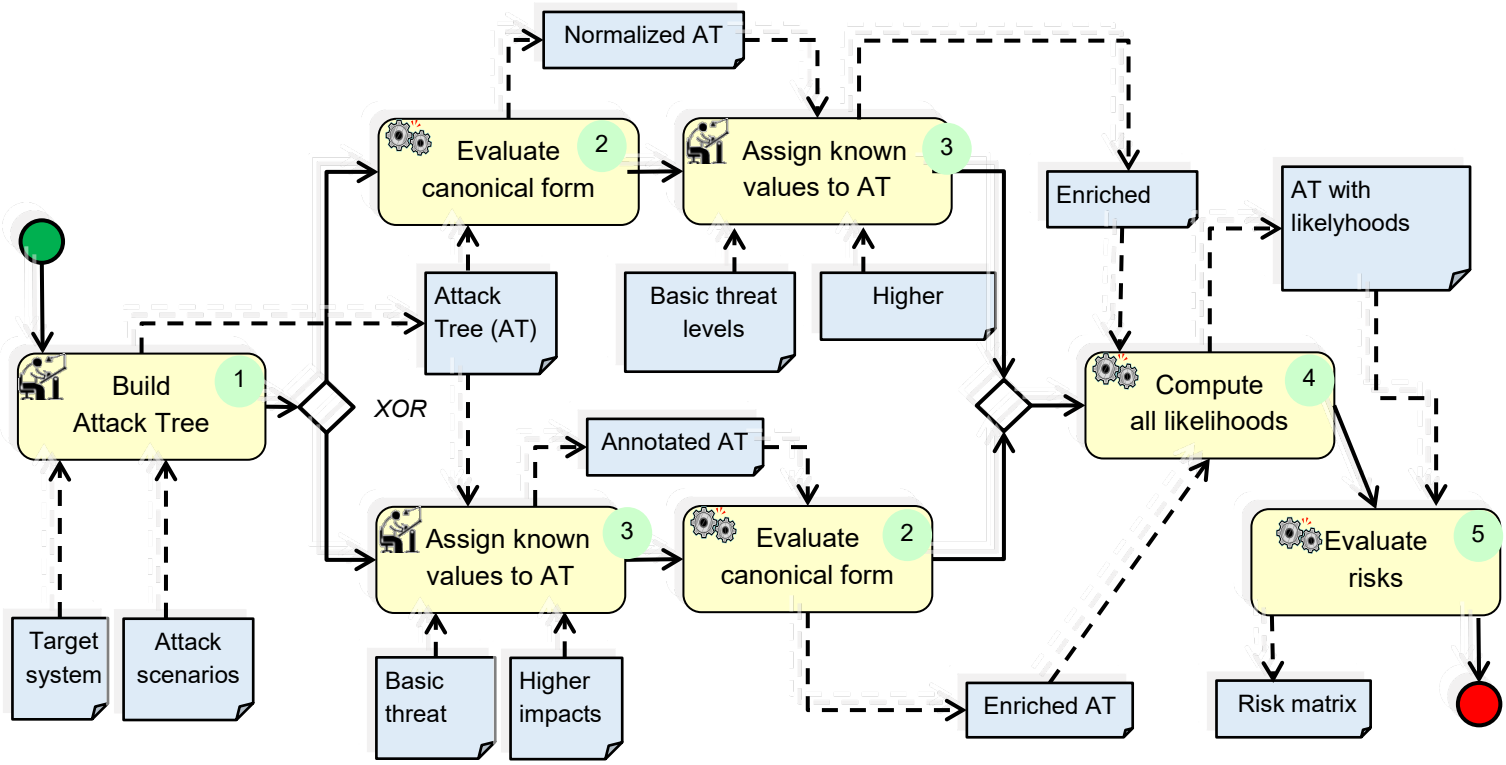


**Figure 40: A pattern for Security Risk Assessment**

The definitions from the pattern include:

- Attack Tree (AT): a tree graph with inner nodes representing logical OR/AND/SAND operations of lower level tree elements
- Basic Attack (BA): bottommost AT nodes, representing atomic or elementary security threats
- Higher Attack State (HAS): all not BA-nodes in the AT.
- Impact (Strength) / Severity Level: severity of consequences of a successful attack
- Likelihood / Threat Level: probability of occurrence of a certain threat or attack
- Risk: Impact * Likelihood
- Target System (TS): the CPS for which security risks shall be analyzed

Participants and important artefacts include:

- System Expert: a person that has sufficient knowledge about the TS for providing all relevant information to the Security Expert
- Security Expert: a person who can identify all BAs relevant for the TS, and who knows how they interrelate for building the AT.
- Target System Description: information at sufficient detail about the TS that can be used by the experts for building the appropriate AT.
- Attack scenarios: considering known attack scenarios help avoiding omission of security threats.
- Attack Tree: tree-like graph representing derivation of complex threats from basic attacks.
- Basic Threat Levels: likelihoods of BAs.
- Higher Impacts: impacts of higher nodes in the BA tree (that are known a priori).
- Normalized AT: AT turned into disjunctive normal form (DNF).

- Enriched AT: normalized AT, annotated with basic threat levels and higher impacts.
- AT with Likelihoods: enriched AT with likelihoods computed for all nodes.
- Risk Matrix: matrix showing the risks of all attacks for all target system. components/items.

Actions and collaborations include:

(1) Build Attack Tree:
   - Identify the BA that are principally possible for the given TS.
   - Multiple attack scenarios are considered by combining BAs via 'OR', 'AND', and 'SAND' (sequential AND) nodes, reflecting experts knowledge about their combinatorial characteristics.
   - Multi-stage attacks are considered by iterating step 2, including new nodes, in a tree-like manner until a topmost attack threat is reached.

(2) Evaluate canonical form: For considering OR combination of attacks, evaluate the canonical form (DNF – Disjunctive Normal Form) of the attack tree. In canonical form all the attack paths are related to an attack state with OR (disjunction) relations. Now, each attack path consists of several basic attack steps related with (S)AND ((sequential) conjunction). This means the attack path will be successful if and only if all the basic attack steps involved are successful.

(3) Assign known values to AT: assign ordinal (e.g. "very low" 0 … "very high" 3) threat levels to BAs', and quantitative severity levels to all higher nodes of the AT.

(4) Compute all likelihoods: For the higher level and multi-level attacks represented (HAS) by the non-leave nodes of the AT, their likelihoods are computed as follows:
   - For OR nodes, the canonical form gives all attack paths to each state related with OR (disjunction) and we do the risk assessment for each attack paths individually. Note that the resulting risks are propagated separately to the next higher level.
   - For AND nodes, likelihood is the minimum of all direct lover level nodes.
   - For SAND nodes, compute likelihood as for AND nodes (i.e. do not consider the impact of sequential dependences on threat level). (The reachability score for following attacks can be taken as preceding attack, as this provides a gateway to the following attacks.)
   - Note: a OR-HAS will be compromised if any of the its attack paths is successful.

(5) Evaluate risks (risk matrix): For each node, compute risk as product likelihood*impact. Represent the result in a matrix with all TS components on one axis, and all threats – basic as well as higher and multi – on the other.

# 9. Appendix B: Pattern for Implementing Self-Healing by Structural Adaptation (SHSA)

SHSA is a method for finding a component to replace the failed component automatically by another component that can provide the functionality of the failed component at least to an acceptable degree.

**Preconditions:**

- The system (CPS) shall be composed of components.
- The system shall be dynamically reconfigurable, i.e. it must be possible to replace components by others at runtime, including redirection/rearranging of communication without interrupting the system's operation.
- To identify an appropriate substituting component, a knowledge base is needed within the system, containing all information about its components with respect to input, output, and functionality.

**Considerations/Constraints:** Usually, three categories of components are distinguished: sensor-oriented, computation-oriented, actuation-oriented ones. In principle, the described approach is appropriate to handle failures of all kinds of components. However, it is unlikely that implicit redundancy exists for actuation components in a system; hence, the described pattern is less appropriate for that category of components.

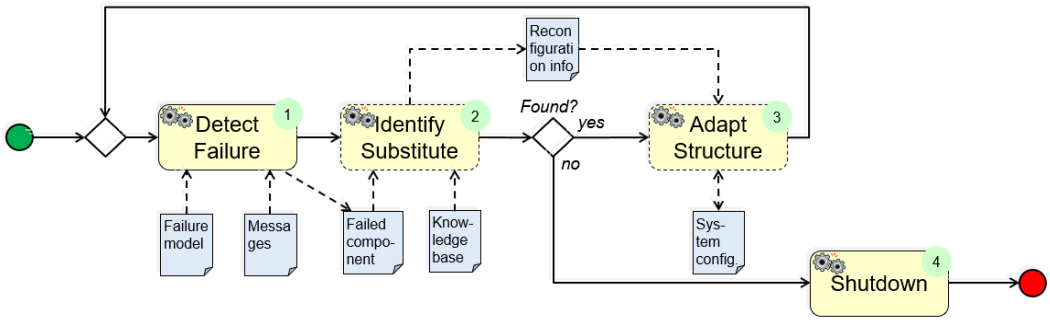


**Figure 41: High-level pattern for SHSA**

**Solution:** Three tasks must be performed in sequence: (i) detects failures, (ii) decides how to replace the failed component, (iii) performs the system reconfiguration. Whether these are realized by a single unit (component), by several ones, or by the whole system in a distributed manner, is of less relevance or, more precise, up to the implementation. Therefore, we concentrate on the activities rather than on (software) components.
The most important property of a component considered here is its output, i.e. the data it provides to other components. Such data have certain properties such as semantics (meaning), value range, precision, and frequency of delivery.

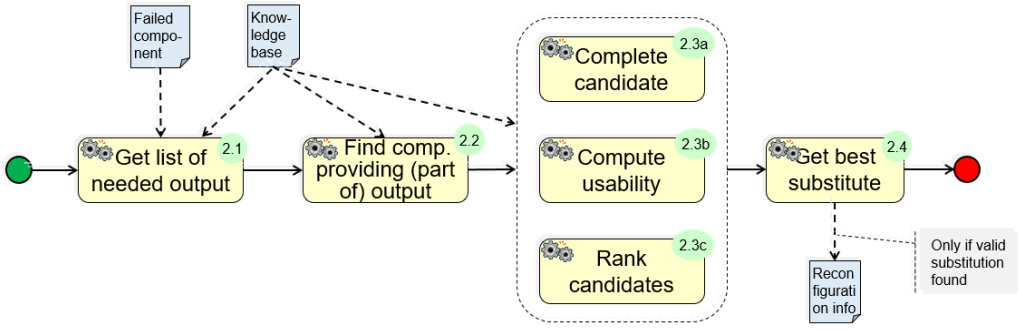


**Figure 42: Identifying a substitute component**

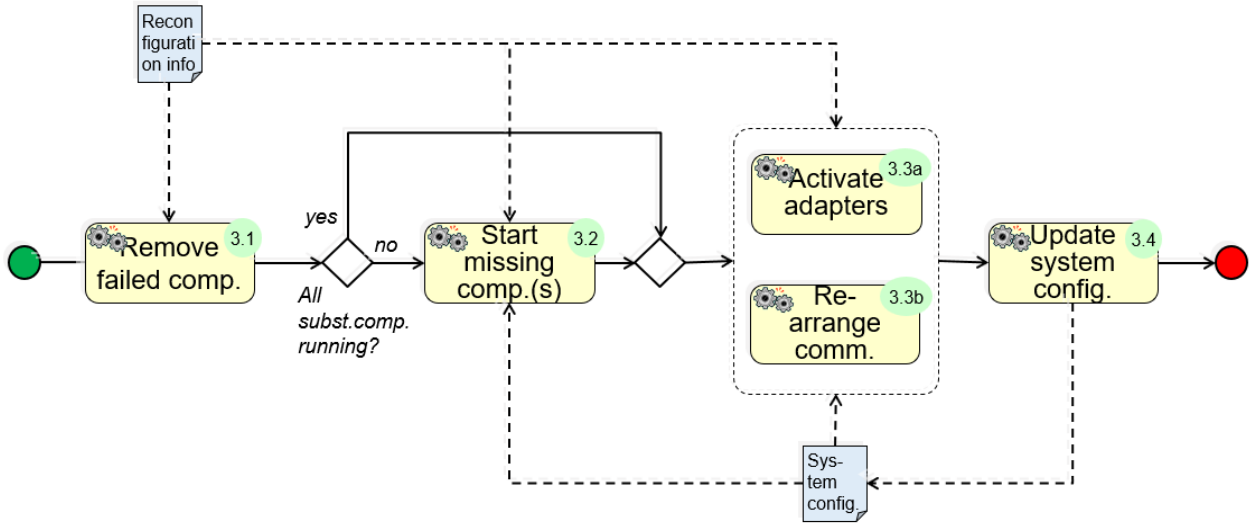


**Figure 43: Adapting the structure of a system**

## Important Artefacts:

***Knowledge base***: machine readable description of all components the target system consists of. For each component, this includes at least its purpose or meaning, required inputs (purpose, value range, units, …) and provided outputs (purpose, value range, units, …). For increasing the flexibility of the substitution finding process, purposes could be interrelated using an ontology. For instance, "brake distance" = ("distance" AND $value \leq dist_{min}$).

***Failure model***: machine-readable specifications of correct behavior for each system component properties describing correctness of their output. If the output deviates from these specifications – perhaps, for a certain timespan, the component is considered as failed. This includes violation of value range as well as delivery frequency. The failure model can be realized as part of the knowledge base.

***Messages***: components communicate by exchanging messages.

***Failed component***: unique identifier of the component that deviates from specification.

***System configuration:*** specification of the currently active components and their properties, e.g. resource consumption and controlled hardware.

***Reconfiguration info***: describes which components to shut down and to start (where), which messages are to be replaced by others, and how to reallocate resources.

## Activities:

1. ***Detect Failure***: The system (e.g. some special unit or component, see "Solution" above) continuously observes the communication between the components and detects deviations from specifications, both in value range and time range. The latter may range from not sending any messages anymore (fail-silent) or too many (babbling idiot). In addition, resource consumptions (e.g. CPU time, memory or power consumption), can by monitored. Whenever a deviation is detected (for a minimum, system-dependent time span,) the *failed component* is forwarded to the next activity *Identify Substitute*.

2. ***Identify Substitute***

   2.1. ***Get list of needed output***: in the *knowledge base*, the output of the *failed component*, i.e. the messages sent by it, are identified.

   2.2. ***Find components providing (part of) output***: in the *knowledge base*, components are searched that provide either the same messages, at least part of them, or messages that can be used as surrogates

for the lost ones. (See the remark on ontology under the *knowledge base* description.) These components are referred to as 'candidates' in the following steps.

2.3. ***Complete and evaluate candidates for substitution***: for each component found in step 2.2, several further steps need to be done. Note that these could be done in sequence, but as it makes sense to stop and eliminate a candidate whenever one of the steps fail, they are shown as a group of activities.

2.3.a. ***Complete candidate***: if a candidate component does not provide all the lost messages, the *knowledge base* is interrogated to find other components that would provide these messages. If this quest fails, the respective component is removed from the list of candidates.

2.3.b. ***Compute usability***: If a component (set) is found that provides the same lost messages, its usability is computed. This means it is evaluated how precise the missing messages are provided, whether transformations are needed to provide them in the expected formatted, how often they are provided, which resources the surrogate components require, including sensors that are perhaps not available, and so on.
If an unavailable resource is needed or another reason opposes the acceptance of the component (set) as substitute, it is removed from the list of candidates.

2.3.c. ***Rank candidates***: if more than one surrogate candidate remains, remaining candidates are ranked, with ranking criteria depending on the target system or the purpose of the failed component, respectively. As result, the chosen candidate is forwarded to next activity.

2.4. ***Get best substitute***: if a valid substitute is found, the *Reconfiguration info* is produced. It contains the identifier of the substitute component(s) together with information whether they must be started, which resources additional resources are required, as well as system dependent data such as, e.g. scheduling details. Finally, if a message is not provided in the expected format, it is specified how the surrogate messages need to be adapted.

3. ***Adapt Structure***: using the *Reconfiguration info*, the system structure is modified to operate with the substitute components instead of the failed one.

3.1. ***Remove failed component***: any resources of the failed component are freed.

3.2. ***Start missing component(s)***: if there are components of the substitution are not yet running, they are started, allocating the resources as specified in the *Reconfiguration info*.

3.3. ***Modify messaging***:

3.3.a. ***Activate adapters***: if a message is not provided exactly in the way as that substituted by it (e.g. different format or dimension), an adapter is needed that converts it into the target shape. Depending on the system architecture, such adapters can be established as output filters of the sending component, input filter of the receiving component, or as transformer in the communication network.
Not that this also applies to any messages needed by newly started components.

3.3.b. ***Rearrange communication***: if necessary, the intra-system network is updated so that the substitute messages receive the right components (as well as the substitute components receive all messages they need).

3.4. ***Update system configuration***: finally, the system configuration data is updated to correctly reflecting the new arrangement of resources, components, and messages.

4. ***Shutdown***: if no substitute could be found, the system has either to be stopped in a safe manner or, if this is not possible, to be turned into a mode of graceful degradation.

## 10. References

[1] Frederic Vannieuwenborg, Sofie Verbrugge and Didier Colle. “Choosing IoT-connectivity? A guiding methodology based on functional characteristics and economic considerations”. Transactions on Emerging Telecommunications Technologies(2018;29:e3308)

[2] Elkhodr, Mahmoud, Seyed A. Shahrestani, and Hon Nin Cheung. "Emerging wireless technologies in the internet of things: a comparative study." International Journal of Wireless and Mobile Networks 8.5 (2016): 67-82.

[3] Tardy, Isabelle, et al. "Comparison of wireless techniques applied to environmental sensor monitoring." SINTEF Report (2017).

[4] Raza, Usman, Parag Kulkarni, and Mahesh Sooriyabandara. "Low power wide area networks: An overview." IEEE Communications Surveys & Tutorials 19.2 (2017): 855-873.

[5] "MulteFire™ lights up the path for universal wireless service." https://www.multefire.org/wp-content/uploads/2016/10/72-multefire-lights-up-the-path-for-universal-wireless-service.pdf. Accessed 4 Dec. 2018.

[6] Comparing IoT Technologies at a Glance | Wi-SUN Alliance." https://www.wi-sun.org/news/comp-iot-tech/. Accessed 4 Dec. 2018.

[7] Long Range Wireless IoT | 2018 Guide to LoRa and Other LPWAN ...." https://www.postscapes.com/long-range-wireless-iot-protocol-lora/. Accessed 4 Dec. 2018.

[8] "Narrowband IoT - Wikipedia." https://en.wikipedia.org/wiki/Narrowband_IoT. 4.12.2018.

[9] "Cellular IoT – Part 9 – 50.000 devices per cell – WirelessMoves." 18 Sep. 2016, https://blog.wirelessmoves.com/2016/09/cellular-iot-part-9-50-000-devices.html. 4 Dec. 2018.

[10] Naik, Nitin. "LPWAN technologies for IoT systems: choice between ultra narrow band and spread spectrum." 2018 IEEE International Systems Engineering Symposium (ISSE). IEEE, 2018.

[11] Mikhaylov, Konstantin, Juha Petaejaejaervi, and Tuomo Haenninen. "Analysis of capacity and scalability of the LoRa low power wide area network technology." European Wireless 2016; 22th European Wireless Conference; Proceedings of. VDE, 2016.

[12] Chen, Min, et al. "Narrow band internet of things." IEEE Access5 (2017): 20557-20577.

[13] Talwar, Shilpa, et al. "Enabling technologies and architectures for 5G wireless." Microwave

[14] Muhammad A. Iqbal, Oladiran G.Olaleye and Magdy A. Bayoumi,”A Review on Internet of Things (Iot): Security and Privacy Requirements and the Solution Approaches”. Global Journal of Computer Science and Technology: ENetwork, Web & Security - 2016.

[15] Kejun Chen, Shuai Zhang, Zhikun Li, Yi Zhang, Qingxu Deng, Sandip Ray and Yier Jin,”Internet-of-Things Security and Vulnerabilities: Taxonomy, Challenges, and Practice”. Journal of Hardware and Systems Security - 10 May 2018.

[16] S. M. Riazul Islam, Daehan Kwak, Md. Humaun Kabir, Mahmud Hossain And Kyung-Sup Kwak,”The Internet of Things for Health Care: A Comprehensive Survey”. IEEE Access - June 1, 2015.

[17] Rashmi Sharan Sinha, Yiqiao Wei and Seung-Hoon Hwang,”A survey on LPWA technology: LoRa and NB-IoT”. Korean Institute of Communication and Information Sciences. March 14, 2017.

[18] Maria Rita Palattella, Mischa Dohler, Alfredo Grieco, Gianluca Rizzo, Johan Torsner, Thomas Engel, and Latif Ladid.” Internet of Things in the 5G Era: Enablers, Architecture, and Business Models”. IEEE Journal on Selected Areas in Communications, Vol. 34, No. 3, March 2016.

[19] The Future of Connectivity in IoT Deployments, Deloite Report.

[20] Richard Harada & Edgar Sotter,”Automated Monitoring for Inspections and Condition-based Maintenance- Part III: Industrial IoT Networks”.

[21] Dong-gil Kim, Seawook Park, Kyungmin Kang and Dongik Lee,” A Deterministic Wireless Network For Feedback Control Based On Ieee 802.15.4”, School of Electrical Engineering and Computer Science, Kyungpook National University 1370 Sankyug-dong, Buk-gu, Daegu, 702-701, Korea - April 21, 2016.

[22] Kay Romer and Friedemann Mattern, “The Design Space of Wireless Sensor Networks”. IEEE Wireless Communications • December 2004.

[23] C.Schmittner, Z. Ma, E. Schoitsch, T. Gruber, „A case study of FMVEA and CHASSIS as safety and security co-analysis method for automotive cyber-physical systems”

[24] Oliver Höftberger. Knowledge-Based Dynamic Reconfiguration for Embedded Real-Time Systems. PhD thesis, TU Wien, Institute of Computer Engineering, Wien, 2015.

[25] D. Ratasich, O. Höftberger, H. Isakovic, M. Shafique, and R. Grosu. A Self-Healing Framework for Building Resilient Cyber-Physical Systems. In 2017 IEEE 20th International Symposium on Real-Time Distributed Computing (ISORC), pages 133-140, May 2017.

[26] D. Ratasich, T. Preindl, K. Selyunin, and R. Grosu. Self-healing by property-guided structural adaptation. In 2018 IEEE Industrial Cyber-Physical Systems (ICPS), pages 199-205, May 2018.
[27] D. Ratasich, F. Khalid, F. Geissler, R. Grosu, M. Shafique, and E. Bartocci. A Roadmap Towards Resilient Internet of Things for Cyber-Physical Systems. IEEE Access, pages 1-24, Jan 2019.
[28] A.Shostack, Threat modeling: Designing for security. John Wiley & Sons, 2014.
[29] IEC. Industrial communication networks - Security for industrial automation and control systems, IEC 62443-4-2. url: https://webstore.iec.ch/publication/34421.
[30] https://satoss.uni.lu/members/piotr/adtool/
[31] https://github.com/IEEM-HsKA/AAD
[32] https://siddhathrokin.wixsite.com/website
[33] Thomas L. Satty: The Analytical Network Process, 2006
[34] [34] CENELEC, European Committee for Electrotechnical Standardization: EN 50126 Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS) – Part 1: Basic requirements and generic process (1999)
[35] CENELEC, European Committee for Electrotechnical Standardization: EN 50128 Railway applications – Communication, signalling and processing systems – SW for railway control and protection Systems (2011)
[36] CENELEC, European Committee for Electrotechnical Standardization: EN 50129 Railway applications – Communication, signalling and processing systems – Safety related electronic systems for signalling (2003)
[37] SAE International: J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (1 2016)
[38] International Electrotechnical Commission: IEC 62443: Industrial communication networks Network and system security
[39] Avizienis, Algirdas, et al. "Basic concepts and taxonomy of dependable and secure computing." IEEE transactions on dependable and secure computing 1.1 (2004): 11-33.
[40] Pooja Mishra; Anil Jaiswal: A Review on Safety Mechanisms in Vehicular Ad hoc Network, 2015
[41] D. Hauer, D. Ratasich, L. Krammer and A. Jantsch, "A Methodology for Resilient Control and Monitoring in Smart Grids," 2020 IEEE International Conference on Industrial Technology (ICIT), Buenos Aires, Argentina, 2020, pp. 589-594, doi: 10.1109/ICIT45562.2020.9067283.
[42] Cominelli, Marco, Paul Patras, and Francesco Gringoli. "Dead on Arrival: An Empirical Study of The Bluetooth 5.1 Positioning System." Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization. 2019.
[43] Demmel, Franz. "Practical aspects of design and application of direction-finding systems." Classical and modern direction-of-arrival estimation. Academic Press, 2009. 53-92.
[44] Banuprakash. R, H. Ganapathy Hebbar, Sowmya. M, Swetha. M, Evaluation of MUSIC algorithm for DOA estimation in Smart antenna, International Advanced Research Journal in Science, Engineering and Technology, vol. 3, Issue 5, pp 185- 188, May 2016
[45] Rares, Buta, et al. "Experimental evaluation of AoA algorithms using NI USRP software defined radios." 2018 17th RoEduNet Conference: Networking in Education and Research (RoEduNet). IEEE, 2018.