An Interpretation and Implementation of Automotive Hardware SPICE

Jaka Ivančič[1], Andreas Riel[2,3(✉)], and Damjan Ekert[2]

[1] Kolektor Group s.o.o, 5280 Idrija, Slovenia
jaka.ivancic@kolektor.com
[2] ISCN GmbH, 8010 Graz, Austria
andreas.riel@grenoble-inp.fr, {ariel,dekert}@iscn.com
[3] Grenoble Alps University, Grenoble INP, G-SCOP, CNRS, 38031 Grenoble, France

Abstract. In the context of the Automotive SPICE® standard process reference and assessment models, many works have been published about the standard's possible and actual interpretations in several industrial environments and applications. Very few of them, however, deal with the question of what detailed electronic hardware development practices can or shall achieve, and in which order they should be performed. Particularly in safety-relevant systems, however, hardware design quality is vital. In this paper, we present an interpretation of Automotive Hardware SPICE as it is practiced within a global tier-2 supplier. We also show its compatibility with the Hardware SPICE process reference and assessment model plug-in released by the intacs™ at the end of 2019.

Keywords: Automotive SPICE® · Hardware development · Hardware design · Hardware validation and testing

1 Introduction and Related Work

Automotive SPICE® (ASPICE) is the worldwide standard for a process reference model (PRM) and process assessment model (PAM) for the development of automotive embedded systems [1]. Traditionally focused on automotive systems and embedded software development, the version 3.0 of the standard (released in July 2015) came up with a plug-in concept in order to open the path for extending the ASPICE process model with practices for hardware engineering (HWE) as well as mechanical engineering (MEE) [1]. In the automotive sector, formalizing and assessing hardware development processes and practices has become a key necessity with more and more automotive subsystems becoming safety-relevant and therefore subject to the ISO 26262 [2], the standard for handling the functional safety of electronics and software in road vehicles. The ISO 26262 relies on ASPICE processes being in place for both system, hardware and software development, and extends it most notably with specific clauses and methods defining the state-of-the-art of how to determine the safety integrity level of automotive items, as well as how to achieve them by design.

Hardware engineering has a vital role in this because functional safety, i.e., the absence of unreasonable risk due to malfunctions of electric/electronic subsystems [2],

M. Yilmaz et al. (Eds.): EuroSPI 2020, CCIS 1251, pp. 684–695, 2020.
https://doi.org/10.1007/978-3-030-56441-4_51

starts at the electronic signal level. Software that has been tested successfully with a 100% test coverage will likely produce errors if the underlying hardware gets defect. This is why the intacs™ Working Group "Hardware Engineering Processes" released an initial version of the HWE plug-in process specification in November 2019 [3], strongly based on previous original works performed in the German SOQRATES initiative [4], as well as on [5]. Specific practical implementations of this process reference model extension to ASPICE have not been published so far, although automotive OEMs keep pushing their suppliers towards higher quality hardware development processes.

This article gives an insight into one of the worldwide first fully blown complete interpretation of the HWE process reference and assessment model extension. It elaborates on the overall model, as well as on the key characteristics of the individual processes, as well as their relationships and rationales.

2 Basic Definitions

The definition of fundamental hardware items in our process is based on the fundamental specification given in [3] and depicted in Fig. 1 below.

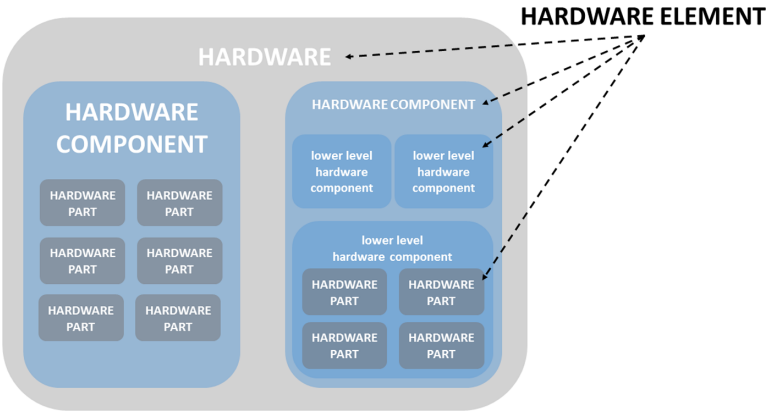

Fig. 1. Graphical specification of hardware elements according to [3]

In our process, "Hardware" is mapped to one or several electronic boards. "Hardware component" is mapped to the item "Hardware Assembly". "Lower level hardware component" is mapped to "Hardware Unit". "Hardware Parts" are considered the basic building blocks of hardware units and are therefore not represented as a dedicated work item. In compliance with the ISO 26262, in our process, a hardware unit is defined as the smallest block of hardware to which a specific hardware sub-function can be assigned. Hardware units are composed of hardware parts, the lowest (indivisible in the application scope) level of hardware (e.g. resistors, capacitors, IC's). Figure 2 shows a power supply hardware unit whose function is to convert

230 V input voltage to a circuit supply of 9 V. This includes different hardware parts and integrates them to a hardware unit implementing a function which can be mapped onto unit testing including e.g. an equivalence class test: Testing the output above 1 A, the fuses F1 shall fire, and there shall be no voltage output. Testing in the 0,8 A ± 0,2 V range, the system operates and delivers current at 9 V. Hardware units can also include diagnose outputs such as a connection to an ADC that writes the measured voltage to a register.

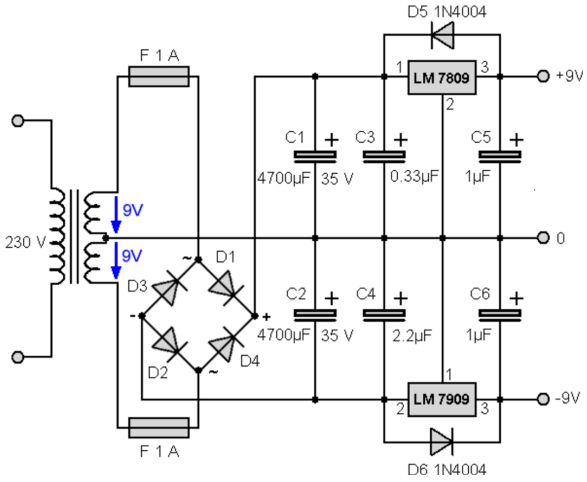

Fig. 2. Example of a hardware unit (detailed schematic design)

Figure 3 depicts the workflow we defined for the *Hardware Unit* work item. The explicit distinction between the schematic design and the layout design for the specification of the hardware unit's detailed design is of particular importance here.

A hardware functional block is defined as a set of hardware units that are related to each other in that they implement a higher functionality on integrated hardware level (e.g., power supply, input-circuit, filtering, micro-controller, driver circuits, etc.). Another example is a power stage for e-motor control which is typically composed of several hardware units implementing functions needed for the power stage (e.g., charge pump, current sensing, temperature sensing). Hardware functional block definitions shall cover the specification of any required hardware units as black boxes with their interfaces and interconnections. In particular, the specification shall cover the hardware block's interface to other functional blocks and units. In Polarion, hardware functional blocks are defined in work items of subtype *HW Functional Block*. The workflow and attributes are the same as for the work item *Hardware Unit*.

A hardware PCB layout represents the integration of any or a particular set of hardware functional blocks and units to a functional hardware circuit. In Polarion, PCBs are represented as work items of type *PCB* with the associated workflow depicted in Fig. 4. A hardware release work item represents the integration of one or

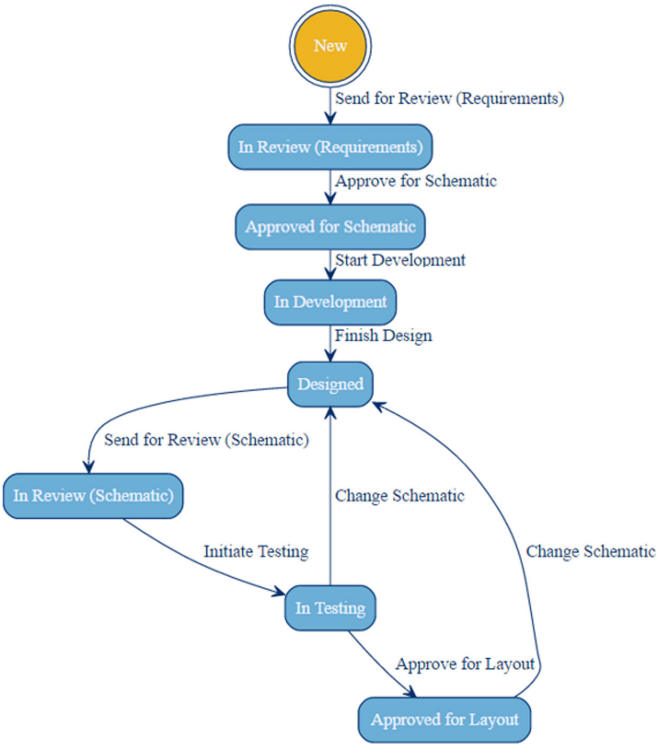

Fig. 3. The hardware unit item workflow

several hardware PCB's and hardware units. The release work item further serves as an interface between teams from development, production, and testing.

The traceability between the different hardware specification levels is facilitated by the Polarion link model and the Polarion work items *Hardware Requirement*, *Hardware Item (subtype Unit or Functional Block), Hardware Release*, and *PCB*, see Fig. 5 below.

Each of these work items includes all the standard field and attributes we defined for system-, software-, hardware-, as well as mechanics-related work items to be managed during their development life cycles in an ASPICE compliant way. Since there are no hardware-related specifics, we will not elaborate on them in this article.

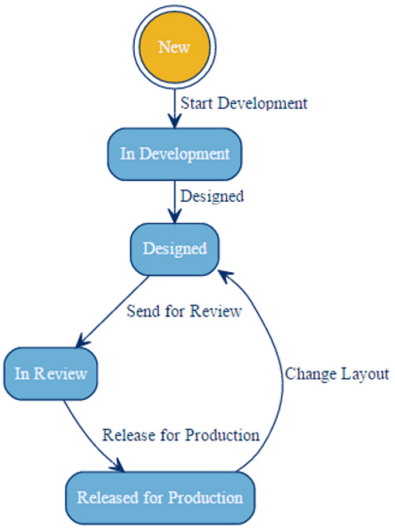

Fig. 4. The PCB item workflow

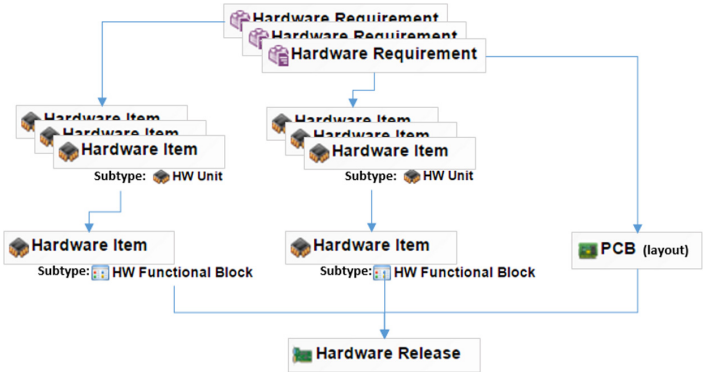

Fig. 5. The hardware work item structure and traceability model in Polarion

3 Hardware Development Process Design

Hardware engineering requires the integration of design, manufacturing and design verification in a way that is different from software engineering (as software does not have to be "manufactured"). In particular, design steps are divided into schematic and layout design, with static and dynamic verification in a simulation environment to be done after each of these two major design steps. The layout design covers both hardware units and hardware functional blocks. Its verification is partly done in a simulation environment before the PCB is actually manufactured and ultimately verified by physical tests.

More specifically, and as shown in Fig. 6, hardware design up to design verification is executed in the following step sequence, and over iterations as required:

1. Create hardware architectural design in the form of a hardware schematic design (of hardware functional blocks).
2. Create hardware detailed design in the form of hardware schematic designs (of hardware units).
3. Perform static/dynamic verification of the hardware schematic designs (schematic design verification through simulation).
4. Create the layout design including any hardware units and hardware architectural design elements (functional blocks).
5. Send the layout design to production (manufacture PCB with all hardware units and functional blocks).

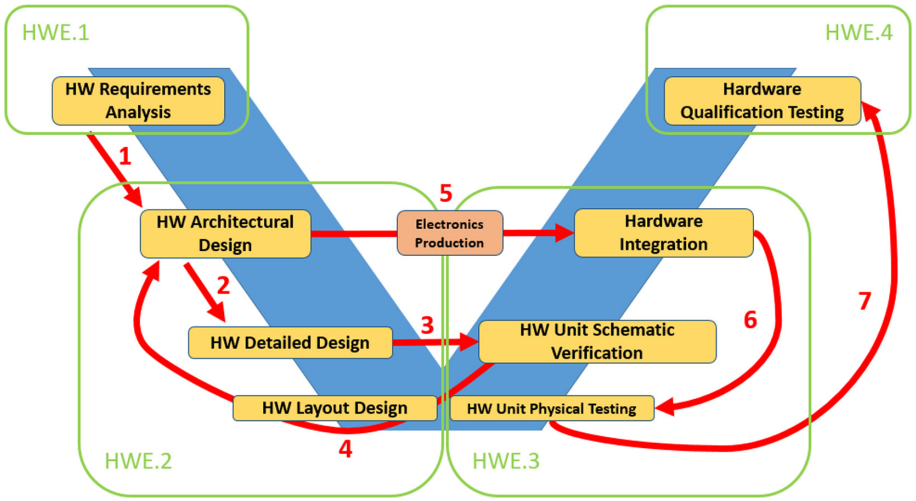

Fig. 6. Integrated hardware design, manufacturing and design verification strategy

6. Perform hardware unit physical tests on the fabricated PCB according to unit requirements.
7. Perform physical hardware qualification testing on the PCB according to hardware requirements.

This sequence of steps clearly shows some specific interpretations of the HWE V-cycle plug-in specified in [3] that we have made:

1. HWE.1 (HW Requirements Analysis) is covered by HW Requirements Analysis.
2. HWE.2 (HW Design) is covered by HW Architectural Design, HW Detailed (Schematic) Design, and HW Layout Design.
3. HWE.3 (Verification against HW Design) is covered by HW Unit Schematic Verification, HW Integration, HW Physical Testing.

4. HWE.4 (Verification against HW Requirements) is covered by Hardware Qualification Testing.

In the following subsections, we will describe the key sub-steps and activities of each of these steps, focusing on those that differ most from related software development process steps.

3.1 HW Requirements Analysis

The purpose of the Hardware Requirements Analysis process is to derive hardware requirements from the system requirements and system architectural design specifications, hardware design standards, as well as internal hardware requirements specifications. Hardware requirements shall define the functional hardware behavior in terms of hardware functions as well as non-functional hardware properties in a solution-neutral way. These hardware requirements subsequently guide the architectural and detailed design of the hardware. We will not describe related process steps and actions further because there is nothing really special compared to software requirements analysis.

3.2 HW Architectural Design

The purpose of the Hardware Architectural Design is to establish the highest level architectural design specification of the hardware to be developed according to the hardware requirements specification. The process identifies which hardware requirements are to be allocated to which hardware items and evaluates the according Hardware Architectural Design against defined criteria. Hardware unit requirements arising from Hardware Architectural Design are mapped to the Hardware Detailed Design level.

The hardware architectural design shall cover the mapping from functional and non-functional requirements to design decisions and hardware items in terms of hardware units and functional blocks. Moreover, it shall specify the internal and external unit and functional block interfaces as well as their interconnections in a way that hardware developers have a very clear and unambiguous specification for the unit detailed design and construction. The specification shall cover both static and dynamic aspects of the hardware architecture as well as specific constraints, solutions and requirements from standards, application guidelines and risk mitigation activities (e.g. dFEMA). Consequently, the following steps are in the scope of hardware architectural design activities:

1. evaluate possible hardware architectural design approaches and justify the choice of the chosen approach,
2. establish a static architecture diagram showing all hardware units and components (set of units) and their signal interfaces and connections,
3. create a hardware unit/functional block work items in Polarion having names corresponding to the names in the diagram and fill out all mandatory attributes,
4. establish dynamic diagrams as considered relevant,

5. consider objectives for resource consumption, reliability, safety, etc. on an architectural level,
6. establish general hardware design decisions as work items of subtype Design Decision in the hardware requirements specification,
7. define relevant criteria for the static hardware verification (e.g., FIT-rates, layout metrics, voltage and temperature levels, etc.),
8. review all the created work products.

Hardware requirements and conceptual architectural design is a basis for static hardware architecture, that consists of:

- identification of hardware units and functional blocks,
- unit functional, non-functional and dynamic requirement specification,
- unit verification criteria,
- interfaces between units and blocks,
- external interfaces,
- applicable PCB layout requirements, mechanical properties or constraints (e.g. positioning of power stage).

Static architecture is given in a graphical representation of units and additional hardware requirements (decisions) that can either become a part of hardware requirements specification or unit requirements specification.

3.3 HW Detailed Design

The purpose of the Hardware Detailed Design process is to establish a detailed design of the hardware units such that they implement and are consistent with the requirements and architectural design. The detailed design specification of these hardware units in terms a documented schematic design is the lowest specification level and therefore provides the basis for the hardware unit construction by developing the hardware layout design. The hardware detailed design and schematic design must implement the hardware requirements in compliance with the hardware architectural design (unit requirements). The hardware unit construction must implement the detailed design in the layout design on the basis of the schematic designs of hardware units, functional blocks, and their interconnections. A standard set of attributes and links are used to support analysis.

The Hardware Detailed Design shall cover the schematic design of the entire hardware based on the hardware architectural design, as well as the hardware requirements. Consequently, the following steps are in the scope of hardware detailed design activities:

1. evaluate hardware schematic design approaches and justify the choice of the chosen approach,
2. establish detailed schematic designs of hardware components, including a detailed specification of component interfaces, as well as the component's units and their interconnections,
3. establish detailed schematic designs of hardware units, including a detailed specification if the unit interfaces and any relevant non-functional static and dynamic

properties (in compliance with related design objectives, e.g. resource consumption, timing behaviour, etc.),
4. establish detailed schematic designs of hardware PCB's in compliance with the hardware architectural design and in full consistency and traceability (by name) to the hardware units and functional blocks,
5. review and release the schematic designs for unit schematic verification.

Schematic design guidelines define which hardware part libraries are allowed, give instructions for creating new components, specify design patterns and rules, as well as naming conventions.

3.4 HW Layout Design

To purpose of the Hardware Layout Design is to develop the hardware units as a hardware layout covering all the units, functional blocks as well as their interconnections on one or several PCB. The hardware layout design covers the entire PCB layout including any unit and functional block. Consequently, the following steps are in the scope of the hardware unit construction activities:

1. per PCB, establish layout designs in compliance with the respective schematic designs for any hardware unit and functional block that shall be integrated on this PCB,
2. apply real-time design rule checking,
3. release the PCB layout design for manufacturing and physical testing.

Just like with the schematic design, PCB layout design guidelines shall assure coherent design practices and decisions throughout the organisation.

3.5 HW Unit Schematic Verification

In order to assure that the design decisions made during the schematic design are correct and to avoid any defects propagating to the layout design and the physical unit and hardware assembly construction, the unit verification steps as described below shall be carried out based on the schematic design and before the PCB layout design. The hardware unit verification shall cover all the hardware units and components assuring their functional completeness (i.e., requirements coverage), correctness (i.e., correct implementation of the requirements before the layout design) and compliance with defined verification criteria. The verification is typically performed by simulation. Consequently, the following steps are in the scope of the hardware unit verification of *functional completeness*:

1. Define test cases with regard to unit requirements and verification criteria resulting in detailed description of test steps and pass fail criteria,
2. set up the simulation environment for test cases,
3. specify tests in the simulation environment,
4. execute these tests,
5. record and analyze the test results,
6. improve the tested units and test cases if required,

7. release the schematic designs for layout design (i.e., unit construction).

These steps aim at verifying hardware units for *correctness* using the defined criteria for verification, and comprise the following activities:

- Review of the hardware unit design.
- Review of the hardware unit specification sheet.
- Review of the hardware unit FIT rate target.
- Integration of the FIT rate and target profile of the hardware unit, including its diagnose capability, in the FMEDA [2, 6, 7] and FTA [8].
- Analysis of the consistency of the assigned ASIL and the determined FIT rate.

Based on this, testing activities comprise circuit simulation in recognized tools, most notably LTspice (by Analog Devices), and based on electronic component simulation models (typically provided by the component manufacturer). Simulation test cases shall include at least the following cases:

- Voltage sweep (verifying operating voltage, continuous and transient voltage levels across the entire circuit),
- Temperature sweep (verifying temperature ranges across the entire circuit),
- Power range check,
- Tolerance calculation (using worst-case analysis, Monte Carlo simulation, etc.).

3.6 HW Physical Testing

Physical hardware unit tests basically involve the same sequence of steps as the static testing. The required activities depend on the hardware item classification according to ISO 26262-5 [2], i.e., either class I (e.g. resistor, capacitor, transistor, diode, quartz, resonator), class II (e.g. fuel pressure sensor, temperature sensor, stand-alone ADC) or class III (e.g. microprocessor, microcontroller, DSP, accelerator) element depending on the item's properties. These classes reflect the difficulty of the verification of the safety-related functionality and the role of the hardware element within the safety concept in safety-relevant systems.

- Class I, II, III: Tables 10–12 in ISO 26262-5 apply.
- Class II and III:
  - Qualification of hardware units for their expected target usage profile.
  - Execution and documentation of the lab tests - a set of electrical tests with measurement points on the ECU layout must be planned to check the function of the hardware unit and its behavior in case of failure conditions. In case of processors performing a low level HIL test with the processor pins as the interface.

It is common that physical unit tests cannot be performed without some level of integration with software or other functional blocks. To overcome this, a higher level testing is allowed as long as the traceability to the unit, as well as a complete unit requirements coverage is assured and maintained.

3.7 HW Integration

The purpose of the Hardware Integration is to integrate and produce (interface to the manufacturing process) the complete hardware following a defined hardware integration checklist to assure the integration of the correct version of all the hardware elements (including required configurations) and to thereby create a defined and documented hardware release. The purpose of the Hardware Integration Test is to verify critical communication interfaces and functional blocks of the produced hardware based on the specified hardware architecture. We will not further elaborate on this process here, since we feel that there are no practices or activities to be particularly highlighted.

3.8 HW Qualification Testing

The purpose of the Hardware Qualification Testing process is to ensure that the integrated hardware is tested to provide evidence for compliance with the hardware requirements and that the hardware is ready for integration into the system. The only thing we want to highlight here is the importance of achieving full coverage of customer-specific hardware qualification/acceptance tests.

Similar to HW Unit Physical tests, it is common that HW Qualification tests are performed on a higher integration level (System Integration or System Qualification) as the testing is not possible without embedded software unless the stimulus is generated by a test setup (e.g. HIL).

4 Conclusion and Outlook

We presented a company-specific interpretation of the Hardware Engineering PRM/PAM of Automotive SPICE. Although for reasons of confidentiality we did not elaborate on the content of the individual processes, we have shown that hardware engineering requires a V-cycle implementation that is different from software engineering. The key differences are in the importance of schematic design for static and dynamic hardware verification, as well as the need of hardware integration on a PCB before the execution of tests on the real hardware from unit test level to qualification test level. This implies a special sequence of running through the V-cycle, as well as clearly defined interface to hardware production on hardware integration level. All the processes along the hardware V-cycle have been defined in detail in terms of process steps, associated roles, activities, and interdependencies. These specifications include special clauses and design patterns for safety-relevant projects that are aligned with the ISO 26262 [7].

The presented process has been applied successfully to several projects and is expected to undergo an ASPICE assessment at the end of 2020. Its key impacts and benefits for our company are

1. an overall improvement of hardware design and development, as well as the quality of related work products;

2. more efficient and complete testing due to well defined requirement-test case pairs complemented with a joint review and common understanding between designer and tester;
3. earlier systematic design error prevention and recognition thanks to a clear link between system requirements, hardware requirements and corresponding design elements (schematics, layout);
4. a more narrowed and comprehensive change, risk and problem resolution management during the implementation of HW units;
5. re-use and cross-project information exchange of hardware requirements, design solutions, test cases as well as information on previous issues, risks and important changes.

In the near future, priority will be given to the improvement of the schematic and layout design guidelines in order to better capitalize on proven design patterns and move towards modular kits and libraries for facilitating variant management. Furthermore, the integration of cybersecurity related practices and aspects is envisaged [9, 10].

References

1. VDA QMC: Automotive SPICE® Process Assessment/Reference Model, Version 3.0 (2017)
2. ISO - International Organization for Standardization: ISO 26262 Road vehicles Functional Safety Part 1-10 (2011)
3. intacs™ Working Group 'HW Engineering Processes': PRM/PAM Hardware Engineering (HWE), Version 1.0 (2019)
4. Schlager, C., Messnarz, R., Sporer, H., Riess, A., Mayer, R., Bernhardt, S.: Hardware SPICE extension for automotive SPICE 3.1. In: Larrucea, X., Santamaria, I., O'Connor, R.V., Messnarz, R. (eds.) EuroSPI 2018. CCIS, vol. 896, pp. 480–491. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-97925-0_41
5. Sporer, H.: Mechatronic system development: an automotive industry approach for small teams. Ph.D. Dissertation at Grez University of Technology (2016)
6. ISO - International Organization for Standardization: IEC 60812 Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA) (2006)
7. Messnarz, R., Kreiner, C., Riel, A., et al.: Implementing functional safety standards (SafEUr). ASQ Softw. Qual. Prof. **17**(3), 4 (2015)
8. ISO - International Organization for Standardization: IEC 61025 Fault tree analysis (FTA) (2006)
9. Messnarz, R., Kreiner, C., Riel, A.: Integrating automotive SPICE, functional safety, and cybersecurity concepts: a cybersecurity layer model. Softw. Qual. Prof. **18**(4), 13 (2016)
10. Macher, G., Sporer, H., Brenner, E., Kreiner, C.: Supporting cyber-security based on hardware-software interface definition. In: Kreiner, C., O'Connor, R.V., Poth, A., Messnarz, R. (eds.) EuroSPI 2016. CCIS, vol. 633, pp. 148–159. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44817-6_12