Comparisation of the software requirements in safety related cases According to IEC 61508

Sigita Andrulyte, Josef Börcsök
Computer Architecture and System Programming
University of Kassel
Wilhelmshöher Allee 71 - 73, 34121 Kassel
GERMANY
sigita@uni-kassel.de, j.boercsoek@uni-kassel.de

Abstract: - The safety of software has become increasingly importance in safety related cases. The importance of choosing the correct safety-level of software will be emphasized in this paper. Each software has to fulfil determined requirements given by the customer or which are specific to a designated sector. The correct selection of a SIL will reduce the number of critical failures, makes the state of development easier and will reduce the cost.

Key-Words: - Software; Safety-Level; IEC 61508; software requirements; safety related software; Safety Integrity Level (SIL)

1 Introduction

Software is used in several different sectors of daily life and should support the users in their free time, at work, etc. However software still causes too many errors, which may lead to huge disasters. As an example several software failures like adjustment and conversion errors and their caused harms will be described more extensively.

Errors occurred at Bank 24 while the change to the Euro. The customers found a thousand billion times higher amounts on their accounts after the first quarter at the accounting. The Bank informed about an individual case and did not inform detailed about the causes of the failure.

In the year 1999 NASA space-probe Mars Orbiter, which had the mission to discover the surface of mars, was lost shortly before reaching the destination. The reason for the failure was detected in the conversion between different units of measurement. The costs, caused by this failure, are specified at 125 billion US-$ [1].

There are numerous similar examples, confirming the importance of software safety. This is especially important for safety critical sectors where software works permanent and without failures, e.g. in the medical sector, flight management systems, etc. [1].

The development process of a safety related product requires methodology, “Know-How”-techniques as well as knowledge about the state-of-the-art. All of these techniques are described in national and international standards according to functional safety. These standards define the requirements for each safety level. All requirements are applied later in the development process.

2 Meaning of SIL

A short presentation of the standard IEC 61508 (last released in the year 2011 –Edition 2) is given in this section.

This standard treats the development of safety related electric/electronic/programmable electronic systems (E/E/PES). The norm describes the basic complete life-cycle of safety related systems and consists of seven parts. Failures of these systems can cause harms on persons, objects and/or the environment. The IEC 61508 is a basic standard and can be used directly or for the creation of additional industrial standards.

In this paper will be no discussion about the whole standard but about part 3 “Software requirements” and is applied in the development of that software, which is part of the safety related system. Development of the software will be done in defined steps. Each step of the software-safety-life-cycle has to be divided into basic operations, whereby field of application, inputs and outputs have to be specified.

Safety-integrity will be valued by the use of a SIL. Safety-integrity specifies the mean probability by which the demanded safety-function will be executed as required under the defined conditions and in the defined time by the safety-related system.

The software-safety-integrity-level (SIL) is one of four levels, which specify a safety-integrity of software inside a safety-related system. Safety-integrity of a software is a value for the probability by which the software will fulfil its safety-function within a period of time. The SIL has the following failure tolerances:

- mean probability of failure on demand of the function (in the operating mode with low rate of demand) or
- probability of a dangerous failure per hour (in the operating mode with continuous rate of demand. [2]

The failure-tolerances are defined in tables 1 and 2. [2]

The target of the standard is to ensure that safety-E/E/PES are working faultless and react correct on their inputs. This is called functional safety. Functional safety is not all, which can guarantee safety. But the IEC 61508 only deals with functional safety. A function, which is executed by a controller to ensure that the system remains in a safe state, is called as a safety-function. Each safety-function defines the safety-targets have to be reached (also called "requirements of a safety-function") and the integrity level by which the safety-function has been implemented (safety-integrity-level).

Table 1 Safety-integrity-level for a safety-function with low rate of demand operating mode

SIL	**Operating mode with low rate of demand** (mean probability of failure on demand)
4	$\geq 10^{-5}$ *to* $< 10^{-4}$
3	$\geq 10^{-4}$ *to* $< 10^{-3}$
2	$\geq 10^{-3}$ *to* $< 10^{-2}$
1	$\geq 10^{-2}$ *to* $< 10^{-1}$

Table 2 - Safety-integrity-level for a safety-function with high or continuous rate of demand operating mode

SIL	**Operating mode with high rate of demand or continuous demand** (probability of dangerous failure per hour)
4	$\geq 10^{-9}$ *to* $< 10^{-8}$
3	$\geq 10^{-8}$ *to* $< 10^{-7}$
2	$\geq 10^{-7}$ *to* $< 10^{-6}$
1	$\geq 10^{-6}$ *to* $< 10^{-5}$

3 Relationship between 61508-2 and 61508-3

The Parts 2 and 3 of IEC 61508 including the requirements for safety lifecycle activities:

- Used during the specification, design, and modification of hardware and software and
- Measures to prevent and/or control of random hardware and systematic errors (the E/E/ PES and software safety lifecycles).

The requirements of the parts 2 and 3 include:

- Usage of measures an technics for classification to correct SIL level and
- Control of systematically failures (including software errors) and random hardware failures.

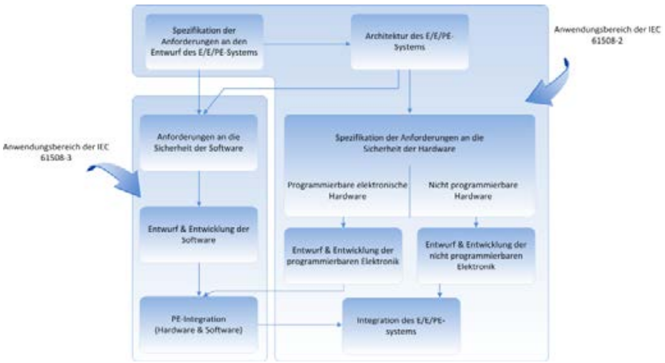
Spezifikation der Anforderungen an den Entwurf des E/E/PE-Systems
Architektur des E/E/PE-Systems
Anwendungsbereich der IEC 61508-2
Anwendungsbereich der IEC 61508-3
Anforderungen an die Sicherheit der Software
Spezifikation der Anforderungen an die Sicherheit der Hardware
Programmierbare elektronische Hardware
Nicht programmierbare Hardware
Entwurf & Entwicklung der Software
Entwurf & Entwicklung der programmierbaren Elektronik
Entwurf & Entwicklung der nicht programmierbaren Elektronik
PE-Integration (Hardware & Software)
Integration des E/E/PE-systems

Figure 1 Relatisonship between 61508-2 and 61508-3

4 Requirements for different SILx level

In appendix of norm 61508-3 are all requirements given for a safety software. These requirements describe a complete software lifecycle. These lifecycle is pictured in Figure [2]. The requirements are grouped into four categories. “++” is much recommended for this SIL level, “+” is recommended, “o” measures or procedure is not recommended and “--“is definitely not recommended for this SIL level. Some interpretations and explanations are in IEC 61508-6, these will not repeat in this paper.

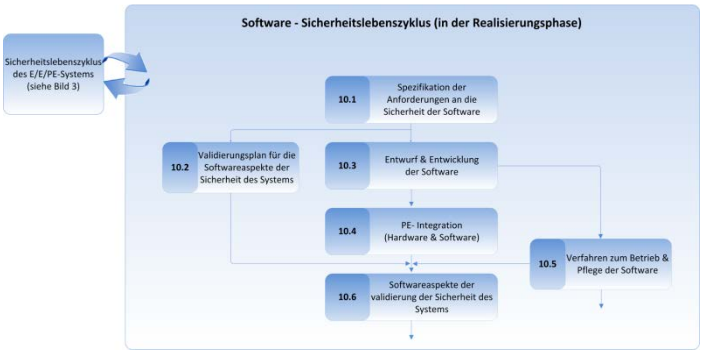
Software - Sicherheitslebenszyklus (in der Realisierungsphase)
Sicherheitslebenszyklus des E/E/PE-Systems (siehe Bild 3)
10.1
Spezifikation der Anforderungen an die Sicherheit der Software
10.2
Validierungsplan für die Softwareaspekte der Sicherheit des Systems
10.3
Entwurf & Entwicklung der Software
10.4
PE- Integration (Hardware & Software)
10.5
Verfahren zum Betrieb & Pflege der Software
10.6
Softwareaspekte der validierung der Sicherheit des Systems

Figure 2

The intense focus in this paper is on requirements of software design and development. For this are detailed requirements given to design software-architecture, utilities and programming languages, detailed design and test of software modules and integration. These requirements were explained twice in second edition of 61508. That the part of software design is explained twice shows how important this part is.

The most requirements of IEC 61503-3 are describe the “detailed design” (DIN EN 61508-3, table A.4) and “test of software modules and integration” (DIN EN 61508-3, table A.5). Different requirements are assigned to different SIL levels. SIL 1 includes only a few requirements and SIL-4 includes the most requirements. SIL 2 is an expansion of SIL 1 and SIL 3 is not so strictly in requirements as SIL 4. It is recommended to use the following requirements for detailed design of SIL3 and SIL4 software: use structured, semiformal or formal design and simplification methods, Computer-aided design tools, defensive programming, modular approach, design and programming guidelines, structured programming etc.

Structured methods are in use to increase the quality of software development in the early phases of lifecycle. The aim of this method is to reach this with exact and intuitive procedures and to define and document requirements and design Features in a logical organization and a structured way. Semi-formal methods like logic-/function block diagram or sequential function charts are responsible for correctness of the program. This is much recommended from SIL3, for systems with lesser security level are semi-formal methods also recommended. Formal methods based on mathematical and suitably for software development. The methods normalize a formal design and formal coding process. This is only highly recommended for SIL4. Defensive programming is recommended from SIL3 and is responsible for development of programs which recognize abnormal flow and data control during the execution. In this case these programs should react in a defined way. For systems with lower security level is defensive programming not so interesting.

A very important Part of software development is the modular approach. This is from SIL1 to SIL 4 much recommended. This approach secures software complexity of bigger software modules. It checks in- and outputs of subroutines and functions. Structured programming is also very important from SIL1 to SIL4. This secures program design and implementation, which could be analyzed without execution. The program may include as little as possible not testable static behavior. For this, the following principles are used:

- Dividing the program into appropriate small software modules, to secure, that this modules are as much decoupled as possible and that all interactions are clearly
- Design of control flow of a program with structured constructs. These are sequence, iteration and selection;
- As few paths as possible during the program modules and a simple relationship between input and output parameters.
- Avoid complicated junctions and especially unconditional jumps (GOTO) in high level languages
- if possible, establish exit conditions of loops on input parameters;
- don’t use complicated conditions for junction and loop conditions [2]

Several requirements which are much recommended for SIL3 and SIL4 are not so important for SIL1-SIL2. Here we can define two groups of security levels, one group with lesser requirements and one group in a life-endangering area.

5 Different in the second edition

In the second edition are many requirements in design and software development.

In the software architecture design part are many recommendations with low priority not included in this edition. Such as regeneration blocks, rules generation and recording executed by repeating sections. The first both are only recommended by each SIL level, the third one is mandatory on the highest SIL level. For software development is now recommended to use monitoring functions, which should serve as a protection against specification and execution errors. There is one more mandatory requirement in SIL4 software development. It should have no state or a minimum number of states, which should limit the complexity of software behavior.

In SIL2-SIL4 should use secure software elements with cyclic behavior with max. Cycle time and which adhere to the time-driven architecture. If secure or verified software elements are available, these elements should help to avoid the need for extensive revalidation or a development of new software for each new application. Cyclic behavior with maximum cycle time and time-controlled architecture helps safety-critical real-time systems for transparent fault tolerance implementation. In SIL 3 and SIL 4 are emphasized the traceability, which has major impact in the design phase. More precisely, is the meaning of traceability explained.

Most requirements which are marked with “--“are not in the new edition. Now only the most important requirements remain, which can guarantee the safety.

6 Conclusion

The result of the comparison is shown in Table III.

The table above shows the individual required measures in development, production and operation of a safety integrated system to achieve a specific SIL for these systems are compared to each other. For this purpose the table is separated into six columns. The first column lists the specifications for the steps of development and production and the following phase of operation of the system. The following four columns contain the requirements for each specification to achieve the specific designated SIL. The last column informs about the applicability of the specifications, i.e. if the specification must be applied to the hardware, software or both parts of the system. This text will describe the differences between the requirements of the several SIL and points to special requirements of the SIL in special specifications.

The SIL are subdivided into four levels. First level describes components or systems with the lowest security characteristic. Second and third level describe systems that have a higher availability and lower failure-rates. SIL 4 is the highest available level in the scale and has the most requirements in relation to safety for the development and production and operation lifecycle. In most cases SIL 4 is not used in productive environments due to the huge costs of these systems in relation to the advantage for the safety function.

The differences between design requirements and description methods are the use of different style of language. For SIL 1 and SIL 2 the use of natural language is sufficient to describe these specifications, though SIL 3 requires additional formal parts of description, such as mathematical equations for descripting its functionality. To achieve SIL 4 the requirements and design specifications have to be composed in a formal way, e.g. by the use of mathematical equations for the whole function of the system and its software including mathematic arguments.

The second specification point as shown in the Table 3 describes of configuration management systems. The implementation of this configuration management can be the use of an automatic version system for software or a revision management for prototypes. This can be done manually by the developers and service personal. For SIL 1 and SIL 2 this should used in the essential part of the system and to achieve SIL 3 or SIL 4 the whole system, including hardware and software, have to be organised in a configuration management system.

Additionally these systems must work automatically in the development and production phase of the product lifecycle, e.g. for new updates to the software or hardware components.

The use of prototyping techniques is required for SIL 4 and SIL 3, and optional for the lower SIL. By these techniques the developers create several prototypes of the system or software modules and verify its function in comparison to the predefined requirements. This leads to the early detection of development failures and reduces the failure rate in later development phases. The use of software based modelling and simulation tools assists the developers to fulfil these specifications.

The use of structured design methods like flow-charts or other diagrams is required for SIL 3 and above, preferred for SIL 2 and optional for SIL 1. These methods are useful to prevent the developers from making design mistakes, which could lead to malfunctions or cost intensive incorrect prototypes. E.g. flow-charts are often used to find critical parts inside a function like unintentional loops or bottlenecks in the process dataflow.

To minimize failures as early as possible in the development process it is a well-proven measure to verify the design of the hardware and software as soon as possible. To achieve SIL 1 the design verification step can be done by a couple of experts from the project team. These are one or more developers who are specially trained for the task of design verification. For the higher levels this has to be done by the whole project team. This depends on the principle that more reviewing developer may find more failures in the hardware or software design.

For each Safety level it is at least recommended to use techniques of project management. From SIL 2 and higher it is required to use a project management. A project management technique is used in nearly every development and production lifecycle to manage and plan each step in development or production and to reduce failures and costs for the designated system or module. Very often the project management will be executed by a specialised project manager or project management team with only administrative tasks.

An independent evaluation of the designed and produced hardware and software is only necessary for achieving SIL 4, though it is recommended for SIL 3 and optional for the lower SIL. For this specification external organisations or corporations examine the resulting hard and software if these systems and modules fulfil all current criteria of the state-of-the-art.

The specification of data evaluation analysis and corrective measures is required for each SIL. The evaluation of data is analysed and the corrective measures have to take place to ensure the safety function of the system or module.

An analysis method for hardware and software is the statistical analysis. This method can also use automatic tests to evaluate the function of the system or software module with several empirical values. These analysis methods are required for SIL 3 and above and optional for the lower SIL.

Another analysis method for the software of the safety integrated system is the dynamic analysis of the software module. This method is required for each SIL and will be executed on the running software on the system under test conditions. The dynamic test can only be used to analyse the software components of the system.

An independent organisation will test the final product if the designated level is SIL 3 or higher. For SIL 2 the test can be executed by an external department and it is optional for SIL 1. The external organisation or department will certificate the system for the SIL if all requirements are fulfilled, e.g. the confirmation of a development process according to the V-Modell.

The final product will be also tested in the phase of service in the productive environment by an external department in undefined time periods to guarantee the fulfilment of all requirements even through the productive service of the system. For SIL 2 and higher these audits are required and for SIL 1 they are optional. Both parts of the system, i.e. hardware and software, will be tested in these audits.

Finally all SIL require a software and hardware quality management system which is compliant to the ISO 9001. The standard ISO 9001 defines the requirements to the quality management system which is responsible for the fulfilment of a constant product quality during all phases of the product lifecycle. The standard also specifies that the multiple procedures during development and production must be documented in a very detailed way. The ISO 9001 certification for the quality management system will be tested in undefined time periods as well as the product supervision mentioned in the last section.

Each of the specifications and the corresponding processes and steps must be documented by the developers' team to provide the evidence that the product fulfils the requirements for the specified SIL. The documentation must be rolled out with the product for the use in productive environment.

This table only gives a superficial overview over the required measures for the product lifecycle of a safety integrated system. The more detailed description of each specification can be found inside the standard itself.

Table 1 - Comparisation of SIL

Specification	SIL 4	SIL 3	SIL 2	SIL 1	Applicability Hardware (H) Software (S)
Requirements and Design-specifications	Formal (mathematic)	Half-formal (e.g. natural language)	Informal (e.g. natural language)	Informal (e.g. natural language)	H/S
Configuration-management	Complete (automatically for development & production)	Complete (automatically for development & production	Yes	Manual	H/S
Prototyping	Yes	Yes	Optional	Optional	H/S
Structured Design-methods (e.g. flow-charts, relational or transfercharts	Yes	Yes	Preferred	Optional	H/S
Design-verification	Yes (Project team)	Yes (Project team)	Yes (Project team)	Yes (Experts)	H/S
Project-management	Yes	Yes	Yes	Preferred	H/S
Independent technical evaluation	Yes	Preferred	Optional	Optional	H/S
Data-evaluation-analysis and corrective actions	Yes	Yes	Yes	Yes	H/S
Statistical analysis (e.g. automatic testing)	Yes	Yes	Optional	Optional	H/S
Dynamical analysis (e.g. automatic testing)	Yes	Yes	Yes	Yes	S
Independent testing	Yes (By external organisation)	Yes (by external organisation)	Yes (preferred, if executed by external department)	Optional	H/S
Additional product monitoring (e.g. independent test)	Yes (by external department	Yes (by external departme	Yes (preferred, if executed	Optional	H/S

	)	nt)	by external department)		
ISO 9001	Yes	Yes	Yes	Yes	H/S

References:

[1] http://campar.in.tum.de/twiki/pub/Chair/TeachingWs04MedInform/Softwaresicherheit.pdf, Januar 2013

[2] *DIN EN 61508*, VDE, 2011.

[3] J. Clerk Maxwell, *A Treatise on Electricity and Magnetism*, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.

[4] I.S. Jacobs and C.P. Bean, *Fine particles, thin films and exchange anisotropy*, in Magnetism, vol. III, G.T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271-350.