

Application software design *Guideline*

Comments on this report are gratefully received by
Johan Hedberg
at SP Swedish National Testing and Research Institute
mailto:johan.hedberg@sp.se

Quoting of this report is allowed but please remember to state the
source!

Summary

This report is focusing on those parts of IEC 61511 that contain requirements on the software realisation.

This report is one of the results of the research project SafeProd supported by VINNOVA (Swedish Agency for Innovation Systems). More information about the project could be found at www.sp.se/safeprod.

1	Introduction	4
1.1	Purpose	4
1.2	References	4
1.3	Scope	5
1.4	Audience.....	5
2	Definitions and abbreviations	6
3	Activities	10
3.1	Overall Architecture design	10
3.2	Application software safety requirements specification	10
3.2.1	Objectives.....	10
3.2.2	Key activity steps	10
3.2.3	Deliverables.....	10
3.3	Application Software safety validation planning	10
3.3.1	Objectives.....	10
3.3.2	Key activity steps	10
3.3.3	Deliverables.....	10
3.4	Application Software Architecture design	11
3.4.1	Objectives.....	11
3.4.2	Key activity steps	11
3.4.3	Deliverables.....	11
3.5	Application Software Development Support Planning	11
3.5.1	Objectives.....	11
3.5.2	Key activity steps	11
3.5.3	Deliverables.....	11
3.6	Application software development and application module development	12
3.6.1	Objectives.....	12
3.6.2	Key activity steps	12
3.6.3	Deliverables.....	12
4	Deliverables.....	14
4.1	Application Software Safety Requirements Specification	14
4.2	Safety Application Software Development Plan.....	14
4.3	Safety Application Software Validation Plan	15
4.4	Application Software Architecture Design Description.....	15
4.5	Application Program Documentation.....	16

1 Introduction

1.1 Purpose

This aim of this report is to be a support during the software design and give guidelines on important aspects in IEC 61511.

This report is only a guideline. In order to fulfil the requirements related to software design IEC 61511 must be used.

This report is one of the results of the research project SafeProd supported by VINNOVA (Swedish Agency for Innovation Systems). More information about the project could be found at www.sp.se/safeprod.

1.2 References

[1] IEC 61511-1 Functional safety: Safety Instrumented Systems for the process industry sector – Part 1: Framework, definitions, system, hardware and software requirements

1.3 Scope

This document gives guidelines on how to apply those parts in [1] that relates to software design.

The software design is one of the most central parts of the safety life cycle according to [1].

The design and engineering of safety instrumented systems (4) is one of the most central parts of the safety life cycle according to [1]. See figure 1.

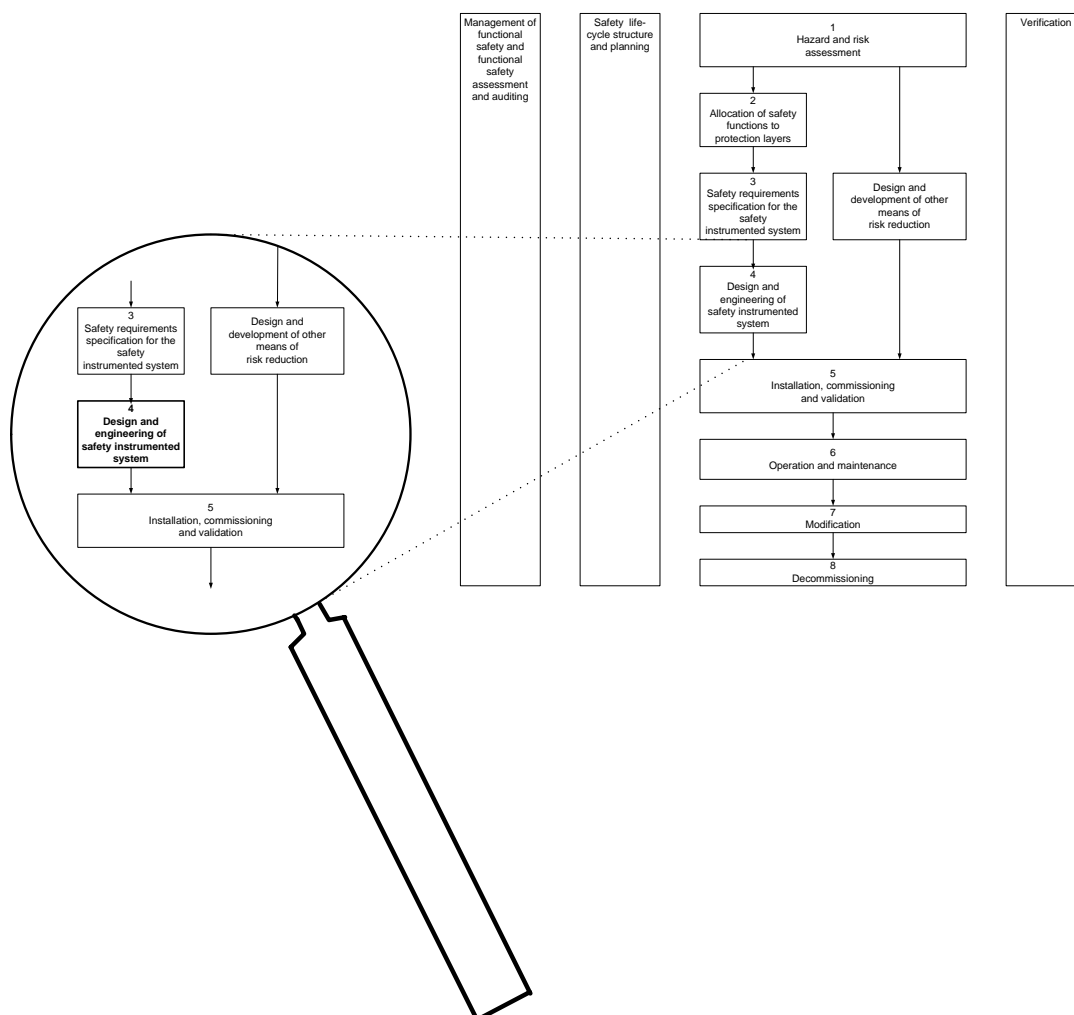


Figure 1. “Design and engineering of safety instrumented system” life-cycle phase in [1]

1.4 Audience

Persons involved in design and engineering of safety instrumented systems.

2 Definitions and abbreviations

basic process control system (BPCS)

system which responds to input signals from the process, its associated equipment, other programmable systems and/or an operator and generates output signals causing the process and its associated equipment to operate in the desired manner but which does not perform any safety instrumented functions with a claimed SIL ≥ 1 (3.2.3 in [1])

component

one of the parts of a system, subsystem, or device performing a specific function (3.2.7 in [1])

continuous mode safety instrumented function

where in the event of a dangerous failure of the safety instrumented function a potential hazard will occur without further failure unless action is taken to prevent it (3.2.43.2 in [1])

demand mode safety instrumented function

where a specified action (for example, closing of a valve) is taken in response to process conditions or other demands. In the event of a dangerous failure of the safety instrumented function a potential hazard only occurs in the event of a failure in the process or the BPCS. (3.2.43.1 in [1])

device

functional unit of hardware or software, or both, capable of accomplishing a specified purpose (for example, field devices; equipment connected to the field side of the SIS I/O terminals; such equipment includes field wiring, sensors, final elements, logic solvers, and those operator interface devices hard-wired to SIS I/O terminals) (3.2.14 in [1])

diagnostic coverage (DC)

ratio of the detected failure rate to the total failure rate of the component or subsystem as detected by diagnostic tests. Diagnostic coverage does not include any faults detected by proof tests. (3.2.15 in [1])

final element

part of a safety instrumented system which implements the physical action necessary to achieve a safe state (3.2.24 in [1])

hardware safety integrity

part of the safety integrity of the safety instrumented function relating to random hardware failures in a dangerous mode of failure (3.2.29 in [1])

instrument

apparatus used in performing an action (typically found in instrumented systems) (3.2.38 in [1]) (3.2.72 in [1]).

logic solver

that portion of either a BPCS or SIS that performs one or more logic function(s) (3.2.40 in [1])

mode of operation

way in which a safety instrumented function operates (3.2.43 in [1])

module

self-contained assembly of hardware components that performs a specific hardware function (i.e., digital input module, analogue output module), or reusable application program (can be portion of a computer program that carries out a specific function (3.2.44 in [1])

non-programmable system

system based on non-computer technologies (i.e., a system not based on programmable electronics [PE] or software) (3.2.47 in [1])

programmable electronics

electronic component or device forming part of a PES and based on computer technology. The term encompasses both hardware and software and input and output units (3.2.55 in [1])

proof test

test performed to reveal undetected faults in a safety instrumented system so that, if necessary, the system can be restored to its designed functionality (3.2.58 in [1])

proven-in-use

when a documented assessment has shown that there is appropriate evidence, based on the previous use of the component, that the component is suitable for use in a safety instrumented system (3.2.60 in [1])

random hardware failure

failure, occurring at a random time, which results from a variety of degradation mechanisms in the hardware (3.2.62 in [1])

redundancy

use of multiple elements or systems to perform the same function; redundancy can be implemented by identical elements (identical redundancy) or by diverse elements (diverse redundancy) (3.2.63 in [1])

safe failure fraction

fraction of the overall random hardware failure rate of a device that results in either a safe failure or a detected dangerous failure (3.2.65.1 in [1])

safety configured logic solver

general purpose industrial grade PE logic solver which is specifically configured for use in safety applications in accordance with chapter 11.5 in [1] (3.2.40.1 in [1])

safety instrumented function (SIF)

safety function with a specified safety integrity level which is necessary to achieve functional safety and which can be either a safety instrumented protection function or a safety instrumented control function (3.2.71 in [1])

safety instrumented system (SIS)

instrumented system used to implement one or more safety instrumented functions. An SIS is composed of any combination of sensor (s), logic solver (s), and final element (s) (3.2.72 in [1])

safety integrity level

discrete level (one out of four) for specifying the safety integrity requirements of the safety instrumented functions to be allocated to the safety instrumented systems. Safety integrity level 4 has the highest level of safety integrity; safety integrity level 1 has the lowest (3.2.74 in [1])

sensor

device or combination of devices, which measure the process condition (for example, transmitters, transducers, process switches, position switches) (3.2.80 in [1])

system

set of elements, which interact according to a design; an element of a system can be another system, called a subsystem, which may be a controlling system or a controlled system and may include hardware, software and human interaction (3.2.84 in [1])

target failure measure

intended probability of dangerous mode failures to be achieved in respect of the safety integrity requirements, specified in terms of either the average probability of failure to perform the design function on demand (for a demand mode of operation) or the frequency of a dangerous failure to perform the SIF per hour (for a continuous mode of operation) (3.2.87 in [1])

undetected/unrevealed/covert

in relation to hardware and software faults not found by the diagnostic tests or during normal operation (3.2.90 in [1])

Abbreviations:

CCF

FMEDA

PFD

SFF

SIL

SIF

SIS

Common Cause Failure

Failure Mode Effects and Diagnostic
Analysis

Probability of Failure on Demand

Safe Failure Fraction

Safety Integrity Level

Safety Instrumented Function

Safety Instrumented System

3 Activities

3.1 Overall Architecture design

Obviously (clause 12.2.2.2 in [1]), the requirements resulting from the SIS architecture should be input to the software safety requirements specification. The SIS Safety requirements should be formulated as of 10.3.1 in [1] and should according to 10.3.2 in [1] be input together with the chosen architecture.

3.2 Application software safety requirements specification

3.2.1 Objectives

- To specify the requirements for the software safety instrumented functions for each SIS function necessary to implement the required safety instrumented functions.
- To specify the requirements for software safety integrity for each safety instrumented function allocated to that SIS.

3.2.2 Key activity steps

- Develop software safety requirements specification
- Requirements reviewed by developers (12.2.2.4 in [1])

3.2.3 Deliverables

- Application Software Safety Requirements Specification

3.3 Application Software safety validation planning

3.3.1 Objectives

- To develop a plan for validating the application software

3.3.2 Key activity steps

- Develop software safety validation plan

3.3.3 Deliverables

-

Safety Application Software Validation Plan

3.4 Application Software Architecture design

3.4.1 Objectives

- To create a software architecture that fulfils the specified requirements for software safety with respect to the required safety integrity level.
- To review and evaluate the requirements placed on the software by the hardware architecture of the SIS.

3.4.2 Key activity steps

- Develop application Software Architecture design document.
- Identify the set of methods and techniques used to develop the application software and justify the use of them (12.4.3.3 in [1])
- Describe and justify the usage of features for maintaining the safety integrity of all data. (12.4.3.5 in [1])

3.4.3 Deliverables

- Application Software Architecture Design Description

3.5 Application Software Development Support Planning

3.5.1 Objectives

- To identify a suitable set of configuration, library management and simulation and test tools compatible with the required safety integrity level, over the whole safety lifecycle of the software (utility software);
- To specify the procedures for development of the application software

3.5.2 Key activity steps

- Develop Safety Application Software Development Plan including the following steps:
 - o Select a suitable list of tools (12.4.4.1 in [1])
 - o Consider the availability of support during the SIS lifetime (12.4.4.2 in [1])
 - o Identify a suitable set of procedures for tool usage (12.4.4.3 in [1])
 - o Eventually justify choice of language (12.4.4.5 in [1])
 - o Specify procedures for usage of the application language. (12.4.4.6 in [1])
- Verify suitability of tools (12.4.4.8 in [1]).

3.5.3 Deliverables

- Safety Application Software Development Plan

3.6 Application software development and application module development

3.6.1 Objectives

- To implement the application software that fulfils the specified requirements for application safety.

3.6.2 Key activity steps

- Application software module design specification (12.4.5.4 in [1])
- Application software module structural test specification (12.4.5.4 in [1])
- Application software development (12.4.5.2, 12.4.5.3, 12.4.5.5 in [1])
- Application software review (12.4.5.6 in [1])

3.6.3 Deliverables

- Application software program (e.g., function block diagrams ladder logic);
- Application program simulation and integration test
-

Application Program Documentation, see section 4.5.

4 Deliverables

4.1 Application Software Safety Requirements Specification

12.2.2.1 in [1] demands this document.

The specification of the requirements for application software safety shall be sufficiently detailed to allow the design and implementation to achieve the required safety integrity, and to allow an assessment of functional safety to be carried out.

The following items shall be included:

1. the functions supported by the application software;
2. capacity and response time performance;
3. equipment and operator interfaces and their operability;
4. all relevant modes of operation of the process as specified in the SIS safety requirement specification;
5. action to be taken on bad process variable such as sensor value out of range, detect loose wire, detected short, etc.;
6. proof tests and diagnostic tests of external devices (e.g., sensors and final elements);
7. software self-monitoring (e.g., includes application driven watch-dogs and data range validation);
8. monitoring of other devices within the SIS (e.g., sensors and final elements);
9. enabling periodic testing of safety instrumented functions when the process is operational;
10. references to the input documents (e.g., specification of the SIF, configuration or architecture of the SIS, hardware safety integrity requirements of the SIS).

Furthermore, this document shall (12.2.2.6 in [1]) provide information allowing proper equipment selection, including

1. functions that enable the process to achieve or maintain a safe state;
2. functions related to the detection, annunciation and management of faults in all SIS subsystems of the SIS;
3. functions related to the periodic testing of safety instrumented functions on-line;
4. functions related to the periodic testing of safety instrumented functions off-line;
5. functions that allow the SIS to be safely modified;
6. interfaces to non safety-related functions;
7. capacity and response time performance;
8. the safety integrity levels for each of the above functions.

4.2 Safety Application Software Development Plan

1. Tools selection specification (12.4.4.1 in [1])
2. Procedures for use of the tools (12.4.4.3 in [1])
3. Eventual implementation language justification (12.4.4.5 in [1])
4. Specify procedures for use of the programming language (12.4.4.6 in [1])

4.3 Safety Application Software Validation Plan

Additional validation planning for the safety application software shall include the following (15.2.2 in [1]):

1. identification of the safety-related software which needs to be validated for each mode of process operation before commissioning commences;
2. information on the technical strategy for the validation including;
 - a. manual and automated techniques;
 - b. static and dynamic techniques;
 - c. analytical and statistical techniques.
3. In accordance with 2., the measures (techniques) and procedures that shall be used for confirming that each safety instrumented function conforms with
 - a. the specified requirements for the software safety instrumented functions (see 12.2 in [1]), and
 - b. the specified requirements for software safety integrity (see 12.2 in [1]);
4. the required environment in which the validation activities are to take place (for example for tests this would include calibrated tools and equipment);
5. the pass/fail criteria for accomplishing software validation including;
 - a. the required process and operator input signals with their sequences and their values;
 - b. the anticipated output signals with their sequences and their values; and
 - c. other acceptance criteria, for example memory usage, timing and value tolerances
6. the policies and procedures for evaluation the results of the validation, particularly failures.

4.4 Application Software Architecture Design Description

The description of the application software architecture design shall (12.4.3.2 in [1]):

1. provide a comprehensive description of the internal structure and of the operation of the SIS subsystem and of its components;
2. include the specification of all identified components, and the description of connection and interactions between identified components (software and hardware);
3. identify the software modules included in the SIS subsystem but not used in any SIF;
4. describe the order of the logical processing of data with respect to the input/output subsystems and the logic solver functionality, including any limitations imposed by scan times;
5. identify all non-SIF and ensure they do not effect the proper operation of any SIF.

4.5 Application Program Documentation

As a minimum, the following information shall be contained in the application program documentation (12.4.2.7 in [1]):

1. legal entity (for example company, author(s), etc.);
2. description;
3. traceability to application functional requirements;
4. logic conventions used;
5. standard library functions used;
6. inputs and outputs; and
7. configuration management including a history of changes.