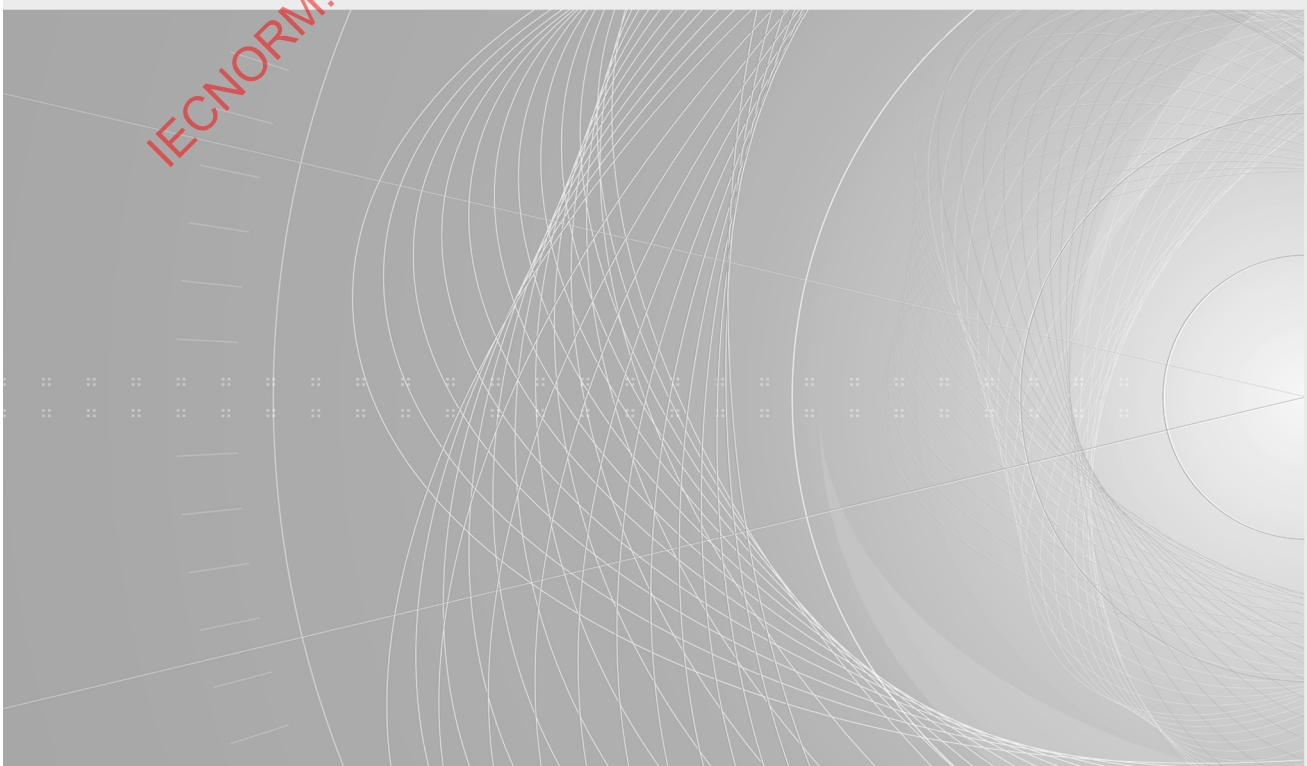
IEC 62138

Edition 2.0 2018-07
REDLINE VERSION

INTERNATIONAL STANDARD

colour inside

Nuclear power plants – Instrumentation and control systems important ~~for~~ to safety – Software aspects for computer-based systems performing category B or C functions



IEC 62138:2018-07 RLV(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - webstore.iec.ch/advsearchform
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing 21 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
67 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

IEC 62138

Edition 2.0 2018-07
REDLINE VERSION

INTERNATIONAL STANDARD

colour inside

Nuclear power plants – Instrumentation and control systems important ~~for~~ to safety – Software aspects for computer-based systems performing category B or C functions

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 27.120.20

ISBN 978-2-8322-5927-6

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

NUCLEAR POWER PLANTS – INSTRUMENTATION AND CONTROL SYSTEMS IMPORTANT ~~FOR~~ TO SAFETY – SOFTWARE ASPECTS FOR COMPUTER-BASED SYSTEMS PERFORMING CATEGORY B OR C FUNCTIONS

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

This redline version of the official IEC Standard allows the user to identify the changes made to the previous edition. A vertical bar appears in the margin wherever a change has been made. Additions are in green text, deletions are in strikethrough red text.

International Standard IEC 62138 has been prepared by subcommittee 45A: Instrumentation, control and electrical power systems of nuclear facilities, of IEC technical committee 45: Nuclear instrumentation.

This second edition cancels and replaces the first edition published in 2004. This edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

a) align the standard with standards published or revised since the first edition, in particular IEC 61513, IEC 60880, IEC 62645 and IEC 62671;
b) merge Clause 5 and Clause 6 of the first edition into a single clause in order to avoid the repetition of the vast majority of the text which proves to be extremely difficult to maintain in consistency;
c) revise clause on the selection of pre-developed software based on experiences from the application of the first edition of the standard on industrial projects. More precise criteria are proposed for the evidence of correctness of pre-developed software;
d) introduce requirements on traceability in consistency with IEC 61513;
e) introduce an Annex A that gives a typical list of software documentation;
f) introduce an Annex B that establishes relationship between IEC 61513 and this document;
g) introduce an Annex C that establishes relationship between IEC 61508 and this document.

The text of this standard is based on the following documents:

FDIS	Report on voting
45A/1201/FDIS	45A/1209/RVD

Full information on the voting for the approval of this International Standard can be found in the report on voting indicated in the above table.

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

In this document, the following print types are used:

- *Requirements and recommendations applicable specifically to class 2 or to class 3 systems appear in italics in Clause 6.*

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The "colour inside" logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this publication using a colour printer.

INTRODUCTION

Structure of the SC 45A standard series – Relationships with other IEC, IAEA and ISO documents

The entry point of the SC 45A standard series is IEC 61513. This standard deals with general requirements for instrumentation and control systems and equipment (I&C systems) that are used to perform functions important to safety in nuclear power plants (NPPs), and structures the SC45A standard series.

IEC 61513 refers directly to other SC 45A standards for general topics related to categorization of functions and classification of systems, qualification, separation of systems, software aspects of computer-based systems, hardware aspect of computer-based systems, control rooms design and multiplexing. The standards referenced directly have to be considered together with IEC 61513 as a consistent document set.

The other SC 45A standards not directly referenced by IEC 61513 are standards related to particular equipment, technical methods or specific activities. Usually, those low level documents, which refer to the documents of the higher levels previously described for the general topics, can be used on their own.

IEC 61513 has adopted a presentation format similar to basic safety publication IEC 61508, with an overall safety lifecycle frame and a system safety lifecycle frame, and provides an interpretation of the general requirements of IEC 61508, parts 1, 2 and 4, for the nuclear application sector. Compliance with IEC 61513 will facilitate consistency with the requirements of IEC 61508 as they have been interpreted for the nuclear industry. In that frame, IEC 60880 and IEC 62138 correspond to IEC 61508, part 3 for the nuclear application sector.

IEC 61513 refers to ISO as well as to IAEA 50-C-QA (now replaced by IAEA 50-C/SG-Q) for topics related to quality assurance.

The SC 45A standards series implements consistently and in detail the principles and basic safety aspects given in the IAEA Code on the safety of nuclear power plants and in the IAEA safety series, in particular the Requirements NS-R-1, "Safety of Nuclear Power Plants: Design" and the Safety Guide NS-G-1.3, "Instrumentation and Control Systems Important to Safety in Nuclear Power Plants". The terminology and definitions used by the SC 45A standards are consistent with that used by the IAEA.

a) Technical background, main issues and organisation of this document

This International Standard provides requirements on the software aspects for computer-based instrumentation and control (I&C) systems performing category B or C functions as defined by IEC 61226. It complements IEC 60880 which provides requirements for the software of computer-based I&C systems performing category A functions.

It is consistent with, and complementary to, IEC 61513:2011. Activities that are mainly system level activities (for example, integration, validation and installation) are not addressed exhaustively by this document: requirements that are not specific to software are deferred to IEC 61513:2011.

This document takes into account the current practices for the development of software for I&C systems, in particular:

– the use of pre-developed software, equipment and equipment families that were not necessarily designed to nuclear industry sector standards;

– the use of application-oriented languages.

b) Situation of the current document in the structure of the IEC SC 45A standard series

IEC 61513 is a first level IEC SC 45A document and gives guidance applicable to I&C at system level.

IEC 62138 is a second level IEC SC 45A document that supplements IEC 61513 concerning software development of computer-based I&C systems performing category B or C functions.

For more details on the structure of the IEC SC 45A standard series, see item d) of this introduction.

c) Recommendations and limitations regarding the application of this document

This document is not intended to be used as a general-purpose software engineering guide. It applies to the software of I&C systems performing category B or C functions for new nuclear power plants as well as to I&C upgrading or back-fitting of existing plants.

For existing plants, only a subset of requirements is applicable and this subset has to be identified at the beginning of any project.

The purpose of the guidance provided by this document is to reduce, as far as possible, the potential for latent software faults to cause system failures, either due to single software failures or multiple software failures (i.e. Common Cause Failures due to software).

This document does not explicitly address how to protect software against those threats arising from malicious attacks, i.e. cybersecurity, for computer-based systems. IEC 62645 provides requirements for security programmes for computer-based systems.

To ensure that this document will continue to be relevant in future years, the emphasis has been placed on issues of principle, rather than specific technologies.

d) Description of the structure of the IEC SC 45A standard series and relationships with other IEC documents and other bodies documents (IAEA, ISO)

The top-level documents of the IEC SC 45A standard series are IEC 61513 and IEC 63046. IEC 61513 provides general requirements for I&C systems and equipment that are used to perform functions important to safety in nuclear power plants (NPPs). IEC 63046 provides general requirements for electrical power systems of NPPs; it covers power supply systems including the supply systems of the I&C systems. IEC 61513 and IEC 63046 are to be considered in conjunction and at the same level. IEC 61513 and IEC 63046 structure the IEC SC 45A standard series and shape a complete framework establishing general requirements for instrumentation, control and electrical systems for nuclear power plants.

IEC 61513 and IEC 63046 refer directly to other IEC SC 45A standards for general topics related to categorization of functions and classification of systems, qualification, separation, defence against common cause failure, control room design, electromagnetic compatibility, cybersecurity, software and hardware aspects for programmable digital systems, coordination of safety and security requirements and management of ageing. The standards referenced directly at this second level should be considered together with IEC 61513 and IEC 63046 as a consistent document set.

At a third level, IEC SC 45A standards not directly referenced by IEC 61513 or by IEC 63046 are standards related to specific equipment, technical methods, or specific activities. Usually these documents, which make reference to second-level documents for general topics, can be used on their own.

A fourth level extending the IEC SC 45A standard series, corresponds to the Technical Reports which are not normative.

The IEC SC 45A standards series consistently implements and details the safety and security principles and basic aspects provided in the relevant IAEA safety standards and in the relevant documents of the IAEA nuclear security series (NSS). In particular this includes the IAEA requirements SSR-2/1, establishing safety requirements related to the design of nuclear power plants (NPPs), the IAEA safety guide SSG-30 dealing with the safety classification of structures, systems and components in NPPs, the IAEA safety guide SSG-39 dealing with the design of instrumentation and control systems for NPPs, the IAEA safety guide SSG-34 dealing with the design of electrical power systems for NPPs and the implementing guide NSS17 for computer security at nuclear facilities. The safety and security terminology and definitions used by SC 45A standards are consistent with those used by the IAEA.

IEC 61513 and IEC 63046 have adopted a presentation format similar to the basic safety publication IEC 61508 with an overall life-cycle framework and a system life-cycle framework. Regarding nuclear safety, IEC 61513 and IEC 63046 provide the interpretation of the general requirements of IEC 61508-1, IEC 61508-2 and IEC 61508-4, for the nuclear application sector. In this framework IEC 60880, IEC 62138 and IEC 62566 correspond to IEC 61508-3 for the nuclear application sector. IEC 61513 and IEC 63046 refer to ISO as well as to IAEA GS-R-3 and IAEA GS-G-3.1 and IAEA GS-G-3.5 for topics related to quality assurance. At level 2, regarding nuclear security, IEC 62645 is the entry document for the IEC SC 45A security standards. It builds upon the valid high level principles and main concepts of the generic security standards, in particular ISO/IEC 27001 and ISO/IEC 27002; it adapts them and completes them to fit the nuclear context and coordinates with the IEC 62443 series. At level 2, regarding control rooms, IEC 60964 is the entry document for the IEC SC 45A control rooms standards and IEC 62342 is the entry document for the IEC SC 45A ageing management standards.

NOTE 1 It is assumed that for the design of I&C systems in NPPs that implement conventional safety functions (e.g. to address worker safety, asset protection, chemical hazards, process energy hazards) international or national standards would be applied.

NOTE 2 IEC SC 45A domain was extended in 2013 to cover electrical systems. In 2014 and 2015 discussions were held in IEC SC 45A to decide how and where general requirement for the design of electrical systems were to be considered. IEC SC 45A experts recommended that an independent standard be developed at the same level as IEC 61513 to establish general requirements for electrical systems. Project IEC 63046 is now launched to cover this objective. When IEC 63046 is published, this NOTE 2 of the introduction will be suppressed.

NUCLEAR POWER PLANTS – INSTRUMENTATION AND CONTROL SYSTEMS IMPORTANT ~~FOR~~ TO SAFETY – SOFTWARE ASPECTS FOR COMPUTER-BASED SYSTEMS PERFORMING CATEGORY B OR C FUNCTIONS

1 Scope

This document ~~provides~~ specifies requirements for the software of computer-based instrumentation and control (I&C) systems performing functions of safety category B or C as defined by IEC 61226. It complements IEC 60880 ~~and IEC 60880-2~~, which provides requirements for the software of computer-based I&C systems performing functions of safety category A.

It is ~~also~~ consistent with, and complementary to, IEC 61513. Activities that are mainly system level activities (for example, integration, validation and installation) are not addressed exhaustively by this document: requirements that are not specific to software are deferred to IEC 61513.

~~IEC 61513 defines the safety classes of I&C systems important to safety as follows:~~

- ~~I&C systems of safety class 1 are basically intended to perform functions of safety category A, but may also perform functions of safety category B and/or C, and non safety-classified functions;~~
- ~~I&C systems of safety class 2 are basically intended to perform functions of safety category B, but may also perform functions of safety category C, and non safety-classified functions;~~
- ~~I&C systems of safety class 3 are basically intended to perform functions of safety category C, but may also perform non safety-classified functions.~~

The link between functions categories and system classes is given in IEC 61513. Since a given safety-classified I&C system may perform functions of different safety categories and even non safety-classified functions, the requirements of this document are attached to the safety class of the I&C system (class 2 or class 3).

~~This standard takes into account the current practices for the development of software for I&C systems, in particular:~~

- ~~the use of pre-developed software, equipment and equipment families that were not necessarily designed to nuclear industry sector standards;~~
- ~~the use of dedicated "black-box" devices with embedded software;~~
- ~~the use of application-oriented languages.~~

This document is not intended to be used as a general-purpose software engineering guide. It ~~provides requirements that~~ applies to the software of I&C systems of safety classes 2 or 3 ~~must meet to achieve system nuclear safety objectives~~ for new nuclear power plants as well as to I&C upgrading or back-fitting of existing plants.

For existing plants, only a subset of requirements is applicable and this subset has to be identified at the beginning of any project.

The purpose of the guidance provided by this document is to reduce, as far as possible, the potential for latent software faults to cause system failures, either due to single software failures or multiple software failures (i.e. Common Cause Failures due to software).

This document does not explicitly address how to protect software against those threats arising from malicious attacks, i.e. cybersecurity, for computer-based systems. IEC 62645 provides requirements for security programmes for computer-based systems.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 60880:2006, *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions*

IEC 61226, *Nuclear power plants – Instrumentation and control ~~systems~~ important ~~for~~ to safety – Classification of instrumentation and control functions*

IEC 61513:~~2001~~ 2011, *Nuclear power plants – Instrumentation and control ~~for systems~~ important to safety – General requirements for systems*

IEC 62671:2013, *Nuclear power plants – Instrumentation and control important to safety – Selection and use of industrial digital devices of limited functionality*

3 Terms and definitions ~~and abbreviations~~

For the purposes of this document, the following terms and definitions ~~and abbreviations~~ apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1
animation
process by which the behaviour defined by a specification is displayed with actual values derived from the stated behaviour expressions and from some input values

[SOURCE: IEC 60880~~-2~~:2006, 3.1]

3.2
application function
function of an I&C system that performs a task related to the process being controlled rather than to the functioning of the system itself

[SOURCE: IEC 61513:2011, 3.1]

3.3
application software
part of the software of an I&C system that implements the application functions

~~NOTE See also System software, Operational system software.~~

Note 1 to entry: Application software contrasts with system software.

Note 2 to entry: Application software is plant specific, so it is not to be considered pre-developed software.

[SOURCE: IEC 61513:2011, 3.2 modified (modified notes to entry)]

3.4
application-oriented language
computer language specifically designed to address a certain type of application and to be used by persons who are specialists of this type of application

Note 1 to entry: Equipment families usually feature application-oriented languages so as to provide easy to use capability for adjusting the equipment to specific requirements.

Note 2 to entry: Application-oriented languages may be used to specify the functional requirements of an I&C system, and/or to specify or design application software. They may be based on texts, on graphics, or on both.

Note 3 to entry: Examples: function block diagram languages, languages defined by IEC 61131-3.

Note 4 to entry: See also general-purpose language.

[SOURCE: IEC 60880:2006, 3.3 modified (addition of note 4 to entry)]

~~**3.5**~~
~~**category of an I&C function**~~
~~one of three possible safety assignments (A, B, C) of I&C functions resulting from considerations of the importance to safety of the functions to be performed. An unclassified assignment may be made if the function is not significant to safety~~

~~(IEC 61513)~~

~~NOTE See also Class of an I&C system.~~

~~**3.6**~~
~~**class of an I&C system**~~
~~one of three possible assignments (1, 2, 3) of I&C systems important to safety resulting from consideration of their requirement to implement I&C functions of differing importance to safety. An unclassified assignment is made if the I&C system does not implement functions important to safety~~

~~(IEC 61513)~~

~~NOTE See also Category of an I&C function.~~

3.5
common cause failure
CCF
failure of two or more structures, systems or components due to a single specific event or cause

Note 1 to entry: Common causes may be internal or external to an I&C system.

[SOURCE: IAEA Safety Glossary, 2016 edition]

3.6
complexity
degree to which a system or component has a design, implementation or behaviour that is difficult to understand and verify

[SOURCE: IEC 61513:2011, 3.9]

3.7
computer program
set of ordered instructions and data that specify operations in a form suitable for execution by a computer

Note 1 to entry: This includes traditional programs written in general-purpose languages. This also includes programs written in application-oriented languages.

[SOURCE: IEC 60880:2006, 3.10, modified (addition of note 1 to entry)]

3.8
computer-based item
item that relies on software instructions running on microprocessors or microcontrollers

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: A computer-based item is a kind of programmable digital item.

Note 3 to entry: This term is equivalent to software-based item.

3.9
configuration management
~~discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control modifications to those characteristics, record and report changes in status, and verify compliance with specified requirements~~

~~(IEC 61513)~~

process of identifying and documenting the characteristics of a facility's structures, systems and components (including computer systems and software), and of ensuring that changes to these characteristics are properly developed, assessed, approved, issued, implemented, verified, recorded and incorporated into the facility documentation

[SOURCE: IAEA Safety Glossary, 2016 edition]

3.10
cybersecurity
set of activities and measures whose objective is to prevent, detect, and react to digital attacks that have the intent to cause:

- disclosures that could be used to perform malicious acts which could lead to an accident, an unsafe situation or plant performance degradation (confidentiality),
- malicious modifications of functions that may compromise the delivery or integrity of the required service by I&C CB&HPD systems (including loss of control) which could lead to an accident, an unsafe situation or plant performance degradation (integrity),
- malicious withholding or prevention of access to or communication of information, data or resources (including loss of view) that could compromise the delivery of the required service by I&C systems which could lead to an accident, an unsafe situation or plant performance degradation (availability).

Note 1 to entry: This definition is tailored with respect to the IEC 62645 scope, focusing on the prevention of, detection of and reaction to malicious acts by digital means on I&C CB&HPD systems. It is recognized that the term "cybersecurity" has a broader meaning in other standards and guidance, often including non-malevolent threats, human errors and protection against natural disasters, which are all out of the scope of IEC 62645.

[SOURCE: IEC 62645:2014, 3.6 modified (removal of note 2 to entry)]

3.11
dedicated functionality
property of devices that have been designed to accomplish only one clearly defined function or only a very narrow range of functions, such as, for example, capture and signal the value of a process parameter, or invert an alternating current power source to direct current. This function (or narrow range of functions) is inherent in the device, and not the product of programmability by the user

Note 1 to entry: Ancillary functions (e.g., self-supervision, self-calibration, data communication) may also be implemented within the device, but they do not change the fundamental narrow scope of applicability of the device.

Note 2 to entry: "Dedicated" in the sense in which it is used in IEC 62671 refers to design for one specific function that cannot be changed in the field.

[SOURCE: IEC 62671:2013, 3.7]

3.12
design specification
document or set of documents that describe the organisation and functioning of an item, and that are used as a basis for the implementation and the integration of the item

3.13
documentation for safety
document or set of documents that specifies how a product can be safely used for applications important to safety

Note 1 to entry: This definition is used in the context of pre-developed software (see 6.3).

3.14
dynamic analysis
process of evaluating a system or component based on its behaviour during execution. In contrast to static analysis

[SOURCE: IEC 60880:2006, 3.15]

3.15
electrical/electronic/programmable electronic item
E/E/PE item
item based on electrical (E) and/or electronic (E) and/or programmable electronic (PE) technology

Note 1 to entry: In this term and its definitions, the word "item" can be replaced by the words: system or equipment or device.

[SOURCE: IEC 61508-4:2010, 3.2.13, modified ("item" added and note to entry modified)]

3.16
equipment family
set of hardware and software components that may work co-operatively in one or more defined architectures (configurations). The development of plant specific configurations and of the related application software may be supported by software tools. An equipment family usually provides a number of standard functionalities (e.g. application functions library) that may be combined to generate specific application software

Note 1 to entry: An equipment family may be a product of a defined manufacturer or a set of products interconnected and adapted by a supplier.

Note 2 to entry: The term "equipment platform" is sometime used as a synonym of "equipment family".

[SOURCE: IEC 61513:2011, 3.17 modified (removal of note 1 to entry)]

3.17
error
discrepancy between a computed, observed or measured value or condition, and the true, specified or theoretical value or condition

Note 1 to entry: See also ~~mistake~~ human error, fault, failure.

[SOURCE: IEC 61513:2011, 3.18, modified (addition of note 1 to entry)]

3.18
executable code
software that is included in the target system

Note 1 to entry: Executable code usually includes instructions to be executed by the hardware of the target system, and associated data.

3.19
failure
~~deviation of the delivered service from the intended one~~
loss of the ability of a structure, system or component to function within acceptance criteria

Note 1 to entry: Equipment is considered to fail when it becomes incapable of functioning, whether or not it is needed at that time. A failure in, for example, a backup system may not be manifest until the system is called upon to function, either during testing or on failure of the system it is backing up.

Note 2 to entry: A failure is the result of a hardware fault, software fault, system fault, or operator or maintenance error, and the associated signal trajectory which results in the failure.

Note 3 to entry: See also ~~mistake~~ human error, fault, ~~failure~~ error.

[SOURCE: ~~IEC 61513~~ IAEA Safety Glossary, edition 2016]

3.20
fault
defect in a hardware, software or system component

Note 1 to entry: Faults may be ~~subdivided into~~ originated from random ~~faults~~ failures, that result e.g. from hardware degradation due to ageing, and may be systematic faults, e.g. software faults, which result from design errors. ~~Random faults result from hardware degradation and cause failures at unpredictable times. Systematic faults result from design errors (for example, software faults) and, in identical conditions, lead systematically to the same failures.~~

Note 2 to entry: A fault (~~in particular~~ notably a design fault) may remain undetected in a system until specific conditions are such that the result produced does not conform to the intended function, i.e. a failure occurs.

Note 3 to entry: See also ~~mistake~~ human error, ~~fault~~ error, failure.

[SOURCE: IEC 61513:2011, 3.21, modified (note 3 to entry modified)]

3.21
firmware
software which is closely coupled to the hardware characteristics on which it is installed. The presence of firmware is generally “transparent” to the user of the hardware component and, as such, may be considered to be effectively an integral part of the hardware design (a good example of such software being processor microcode). Generally, firmware may only be modified by a user by replacing the hardware components (for example, processor chip, card, EPROM) which contain this software with components which contain modified software (firmware). Where this is the case, configuration control of the hardware components of the equipment effectively provides configuration control of the firmware. Firmware, as considered by IEC 60987, is effectively software that is built into the hardware

[SOURCE: IEC 60987:2007, 3.4]

3.22
functional validation
verification of the correctness of the application functions specifications ~~versus~~ against the top level plant functional and performance requirements. It is complementary to the system validation that verifies the compliance of the system with the functions specification

[SOURCE: IEC 61513:2011, 3.23]

3.23
general-purpose language
computer language designed to address all types of usage

Note 1 to entry: The system software of equipment families is usually implemented using general-purpose languages.

Note 2 to entry: Examples: Ada, C, Pascal.

Note 3 to entry: See also application-oriented language.

[SOURCE: IEC 60880:2006, 3.20 modified (note 3 to entry added)]

3.24
human error (or mistake)
human action ~~(or inaction)~~ that produces an unintended result

Note 1 to entry: See also fault, error, failure.

[SOURCE: ~~IEC 60880-2~~ IEC 61513:2011, 3.26 modified (note 1 to entry added)]

3.25
I&C architecture
organisational structure of the I&C systems of a plant which are important to safety

[SOURCE: IEC 61513:2011, 3.27]

3.26
I&C system
system, based on E/E/PE items, performing plant I&C functions as well as service and monitoring functions related to the operation of the system itself

Note 1 to entry: The term is used as a general term which encompasses all elements of the system such as internal power supplies, sensors and other input devices, data highways and other communication paths, interfaces to actuators and other output devices. The different functions within a system may use dedicated or shared resources.

Note 2 to entry: The elements included in a specific I&C system are defined in the specification of the boundaries of the system.

Note 3 to entry: See also the definition of E/E/PE item and the associated notes.

Note 4 to entry: According to their typical functionality, IAEA distinguishes between automation / control systems, HMI systems, interlock systems and protection systems.

3.27
integration
progressive aggregation and verification of components into a complete system

3.28
library
collection of related software elements that are grouped together, but which are individually selected for inclusion in the final software product

[SOURCE: IEC 60880:2006, 3.24]

3.29
mode of ~~behaviour~~ operation
functional state of an item where it provides a specific operational behaviour

~~NOTE~~ EXAMPLE: initialisation mode, normal mode, ~~downgraded~~ degraded modes to be taken in case of error in the item ~~or in its environment~~.

3.30
operational system software
~~part of system~~ software ~~the executable code of which runs~~ running on the target processor during system operation

~~(IEC 61513)~~

~~NOTE 1~~ EXAMPLE: Operating system, input/output ~~and communication~~ drivers, exception handler, ~~scheduler, interrupt management, on-line diagnostic, redundancy and graceful degradation management,~~ communication software, application-software libraries, self-supervision, redundancy and graceful degradation management.

~~NOTE 2 See also Application software, System software.~~

3.31
parameter
data item governing the behaviour of the I&C system and/or of its software, and that may be modified by operators during plant operation

3.32
pre-developed software
software ~~part~~ that already exists, is available as a commercial or proprietary product, and is being considered for use

Note 1 to entry: In this document, pre-developed softwares are divided in two different types:

a) complete operational system software,

b) software components.

Note 2 to entry: Pre-developed software may be divided into software that has not been specifically developed for a specific hardware environment, and software integrated in hardware components that has to be used in association with this hardware.

Note 3 to entry: In this document, this term does not cover software tools, even when they are pre-developed.

Note 4 to entry: Application software is plant specific, so it is not to be considered pre-developed software.

[SOURCE: ~~IEC 61513~~ IEC 60880:2006, 3.28 modified (notes to entry added)]

3.33
pre-existing items
hard- or software or software-based equipment that already exists, is available as a commercial or proprietary product, and is being considered for use

Note 1 to entry: This definition is included for the consistency of the terms and definitions with IEC 61513:2011, but not used. In this document, dedicated to software, the term pre-developed software is used.

[SOURCE: IEC 61513:2011, 3.36 modified (note 1 to entry modified)]

~~**3.25**~~
~~**program**~~
~~document written by a human being that is transformed into executable code by automated tools~~

~~NOTE This includes traditional programs written in general purpose languages. This also includes programs written in application oriented languages.~~

~~**3.26**~~
~~**security**~~
~~capability of a computer-based system to provide adequate confidence that unauthorised persons and systems can neither modify the software and its data nor gain access to the system functions, and yet to ensure that this is not denied to authorised persons and systems~~

~~(IEC 61513)~~

3.34
programmable digital item
item that relies on software instructions or programmable logic to accomplish a function

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: The main kinds of programmable digital items are computer-based items and programmable logic items.

Note 3 to entry: This term used by IEC SC 45A is equivalent to programmable electronic item (PE item) defined according to IEC 61508.

3.35
programmable logic item
item that relies on logic components with an integrated circuit that consists of logic elements with an inter-connection pattern, parts of which are user programmable

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: A programmable logic item is a kind of programmable digital item.

Note 3 to entry: See also the definition of E/E/PE item and the associated notes.

3.36
self-supervision
automatic testing of system hardware performance and software consistency of a computer-based I&C system

[SOURCE: IEC 60671:2007, 3.8]

3.37
software
programs (i.e. sets of ordered instructions), data, rules and any associated documentation pertaining to the operation of a computer-based I&C system

[SOURCE: ~~IEC 60880~~ IEC 61513:2011, 3.51]

3.38
software component
~~one of the design entities that make up a software item. It may be subdivided into other software components~~

~~(IEC 61513)~~

one of the parts that make up a complete software. Software components need to be integrated to form complete software

Note 1 to entry: In this document, a pre-developed software item can be considered a software component only if it is integrated in larger software to form complete operational system software. In particular, verification and validation of the complete operational system software has to be performed with the software components embedded. The integration may be within software that runs on a single processor, for example for Real Time Operating Systems or libraries. The integration may also be within software that run in close cooperation on several processors, for example the firmware of communication modules or input/output modules.

3.39
software development
~~phase~~ all activities of the software lifecycle that lead~~s~~ to the creation of the software of an I&C system or of a software product~~. It~~ and that cover~~s~~ all the ~~activities~~ phases from software requirements specification to validation and installation on site

3.40
software modification
change in an already agreed document (or documents) leading to an alteration of the executable code

Note 1 to entry: Software modifications may occur either during initial software development (for example, to remove faults found in later stages of development), or after the software is already in service.

[SOURCE: IEC 60880:2006, 3.36]

3.41
software safety lifecycle
necessary activities involved in the development and operation of the software of an I&C system important to safety occurring during a period of time that starts with the software requirements specification and finishes when the software is withdrawn from use

[SOURCE: ~~IEC 61513~~ IEC 60880:2006, 3.37]

3.42
software validation
test and evaluation of integrated software to ensure compliance with the functional, performance and interface specifications imposed by the I&C system requirements

Note 1 to entry: In this document, software validation is considered a part of system validation.

3.43
static analysis
process of evaluating a system or component based on its form, structure, content or documentation. In contrast to dynamic analysis

[SOURCE: IEC 60880~~-2~~:2006, 3.40]

3.44
system software
~~part of the software of an I&C system designed for a specific computer or equipment family to facilitate the development, operation and modification of these items and associated programs~~

~~NOTE The system software of equipment families is usually composed of operational system software and of support system software (software tools).~~

software designed for a specific computer system or family of computer systems to facilitate the operation and maintenance of the computer system and associated programs, for example, operating systems, computers, utilities. System software is usually composed of operational system software and support software

Note 1 to entry: Operational system software: software running on the target processor during system operation, such as: operating system, input/output drivers, exception handler, communication software, application-software libraries, self-supervision, redundancy and graceful degradation management.

Note 2 to entry: Support software: software that aids in the development, test, or maintenance of other software and of the system such as compilers, code generators, graphic editor, off-line diagnostic, verification and validation tools, etc.

Note 3 to entry: See also application software~~, Operational system software~~

[SOURCE: IEC 61513:2011, 3.58 modified (notes 2, 3 and 4 to entry added)]

3.45
system validation
confirmation by examination and provision of other evidence that a system fulfils in its entirety the requirement specification as intended (functionality, response time, fault tolerance, robustness)

Note 1 to entry: The 2016 edition of the IAEA Safety Glossary gives the two following definitions:

Validation: The process of determining whether a product or service is adequate to perform its intended function satisfactorily. Validation may involve a greater element of judgment than verification.

Computer system validation: The process of testing and evaluating the integrated computer system (hardware and software) to ensure compliance with the functional, performance and interface requirements.

Firstly, the definition "system validation" is a specific case of validation. It refers to a specific product, namely to the validation of an I&C system. This is consistent with the IAEA definition. Secondly, the IEC definition specifies the reference of validation, namely the requirement specification whereas the IAEA definition only refers to the "intended function".

[SOURCE: IEC 61513:2011, 3.59]

3.46
systematic fault
fault related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors

[SOURCE: IEC 61513:2011, 3.60]

3.47
verification
confirmation by examination and by provision of objective evidence that the results of an activity meet the objectives and requirements defined for this activity

[SOURCE: ~~ISO 12207~~ IEC 61513:2011, 3.62]

~~**3.36**~~
~~**abbreviation**~~
~~**I&C**: Instrumentation and Control~~

4 Symbols and abbreviated terms

CB	Computer-based
CCF	Common cause failure
EPROM	Erasable Programmable Read Only Memory
HMI	Human machine interface
HDL	Hardware Description Language
HPD	HDL-Programmed Device
I&C	Instrumentation and control
NPP	Nuclear power plant

5 Key concepts and assumptions

5.1 General

Clause 5 presents some of the key concepts and assumptions about the nature and the development of the software of I&C systems of safety class 2 or 3, upon which the normative text is based.

5.2 Types of software

Figure 1 illustrates the ~~variety~~ range of services offered by software ~~and software components~~ in a typical I&C system or I&C architecture. Software ~~components~~ may often be defined as being either system software or application software. System software may also be divided

into operational system software, which is embedded in safety classified I&C systems, and support system software (or software tools) which is either off-line or embedded in non-safety classified support systems. Software may also be found in dedicated devices such as sensors and actuators, communication devices and Uninterruptible Power Supplies (UPSs).

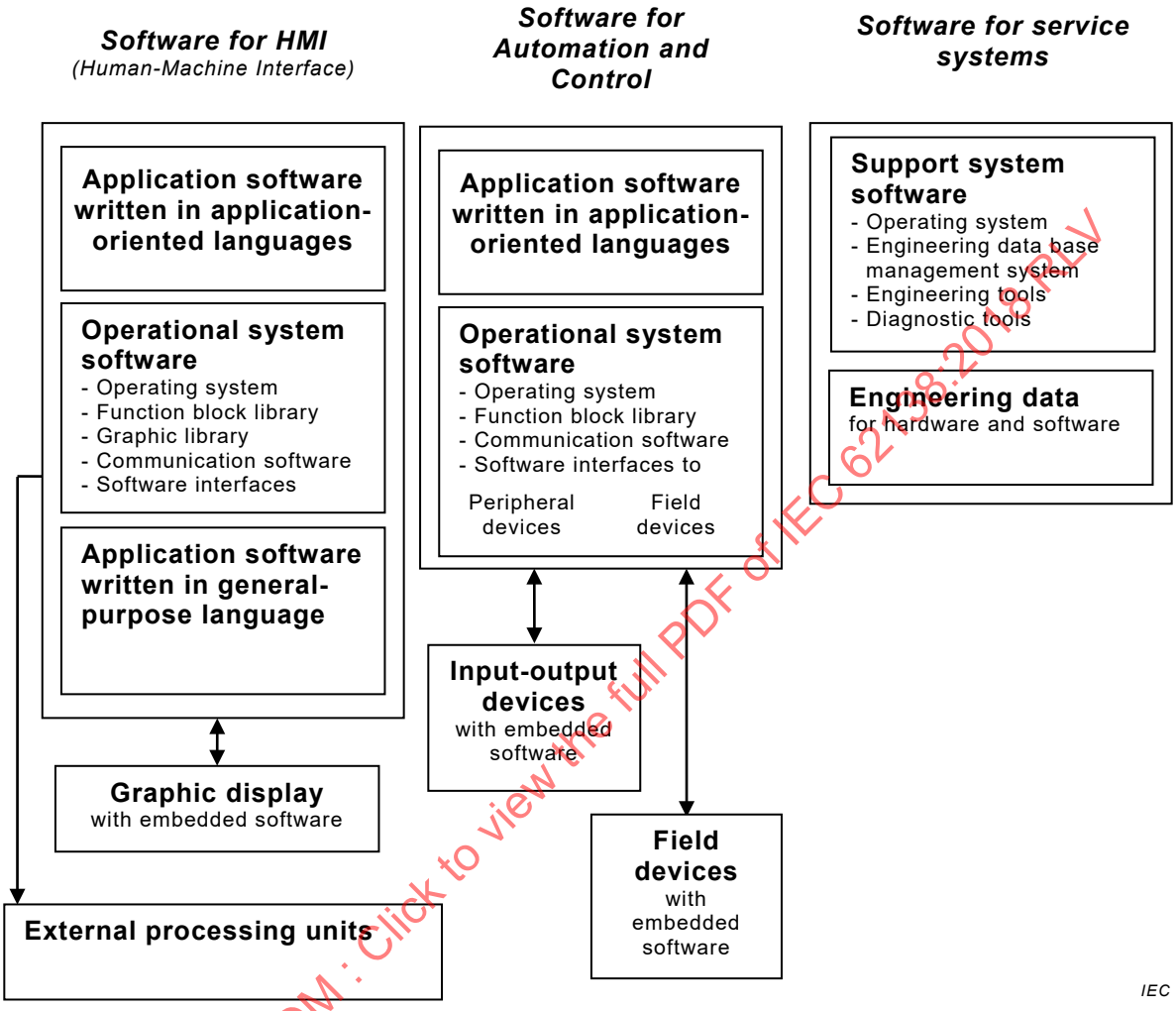


Figure 1 – Typical software parts in a computer-based I&C system~~s~~

The software in an I&C system may also be divided into pre-developed software (which usually provides functions useful to a range of I&C systems) and new software (which is developed to the specific needs of the I&C system). ~~System software is usually pre-developed, and application software is usually new, but this is not an absolute rule.~~ The requirements of this document which address issues that are ~~applicable~~ relevant to new software may also be applied retrospectively to pre-developed software. In some instances, however, this document ~~also~~ provides alternative requirements ~~that may be applied~~ specifically to address issues relevant to pre-developed ~~software or dedicated devices with embedded~~ software.

Many modern equipment families are provided with extensive application-oriented development tools that enable plant or system engineers to specify their requirements using graphical techniques. The tools may automatically translate the ~~graphical~~ graphics representing computer programs into executable application software. When these tools are of adequate quality, this approach is considered to reduce the risk of faults.

5.3 Types of configuration data

Many system designs make extensive use of configuration data. Configuration data may be associated with operational system software or with application software. Configuration data associated with application software consists mainly of plant engineering data resulting from the design of the plant, and is often prepared by plant designers who are not required to have software skills. Configuration data may be divided into:

- data items which are not intended to be modified on-line by plant operators, and which are submitted to the same requirements as apply to the rest of the software;
- parameters, i.e., data items which may be modified by operators during plant operation (for example, alarm limits, set points, data required to calibrate instrumentation) and which need specific requirements.

5.4 Software and system safety lifecycles

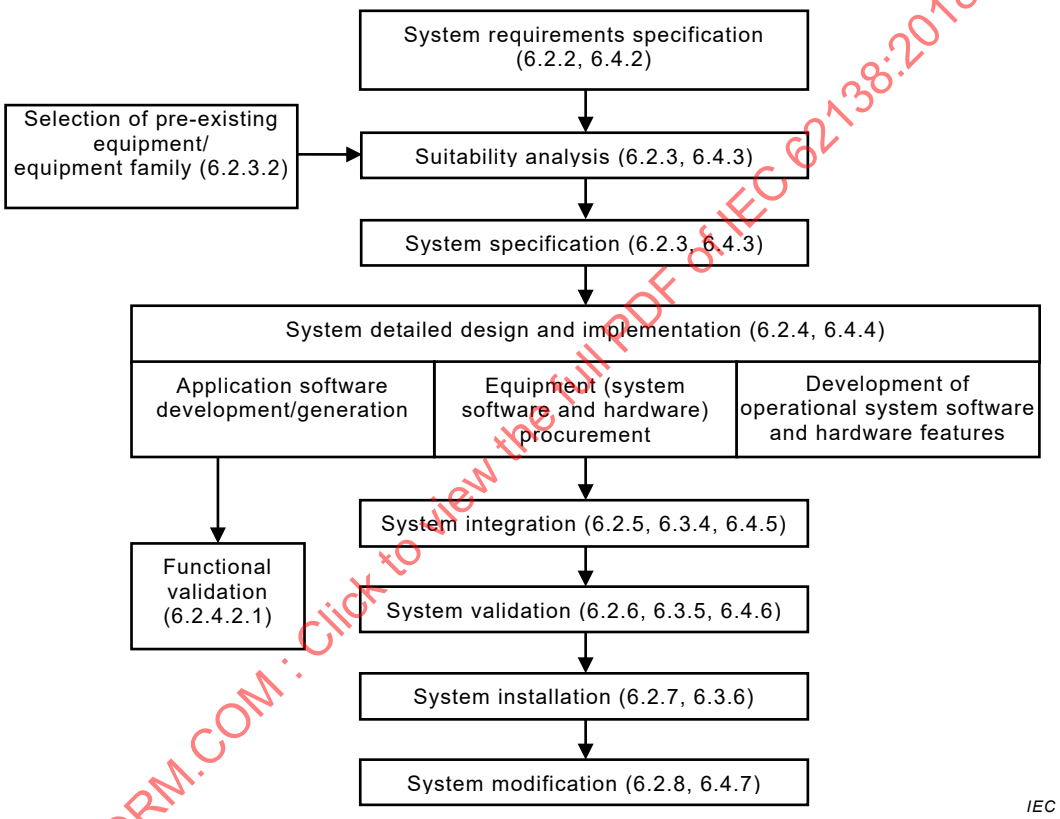
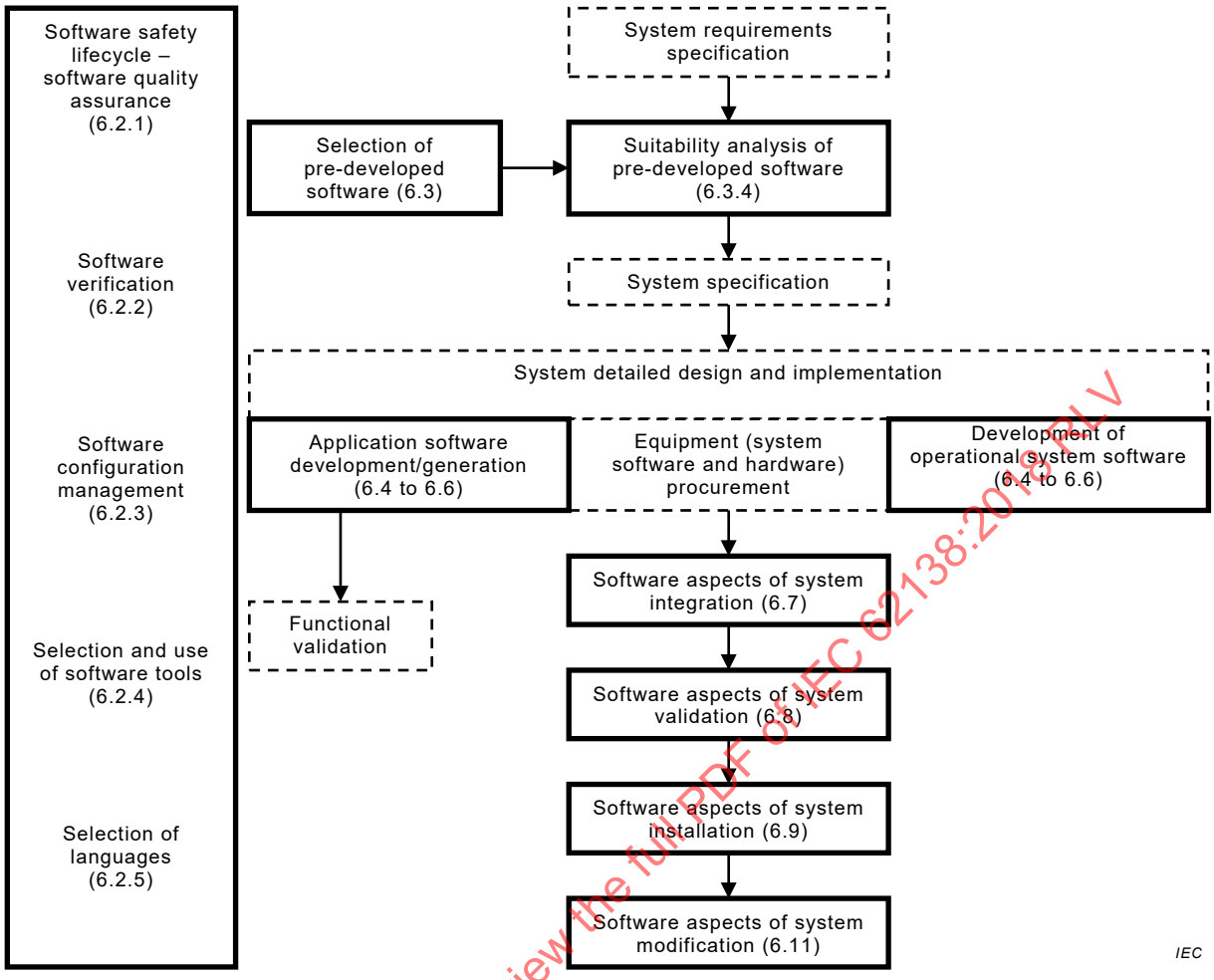


Figure 2 – Activities of the system safety lifecycle (as defined by IEC 61513:2011)

Software usually contributes strongly to the functions performed by the I&C system. It may also support additional functions ~~introduced by~~ needed for the operation of the system ~~design~~ itself (for example, initialisation and ~~surveillance~~ supervision of hardware, communication between, and synchronisation of, sub-systems). Thus, the software safety lifecycle is in most cases strongly integrated with the system safety lifecycle. In particular, the software requirements specification is a part of, or is derived directly from, system specification and system design.

Although the verification of ~~new~~ software ~~components~~ is definitely a part of the software safety lifecycle, there is often no separate and well-identified boundary between software integration and system integration. Therefore, in this document, software integration is considered to be a part of system integration. Software validation too is considered a part of system validation.



NOTE Boxes in thin dotted lines represent system activities not addressed in this document.

Figure 3 – Software related activities in the system safety lifecycle

Figure 2 and Figure 3 illustrate the relationship between the activities of the software safety lifecycle and the activities of the system safety lifecycle.

It ~~should~~ has to be noted that although IEC 61513:2011 identifies two different paths for the implementation of ~~new~~ software (application software and operational system software, see Figure 2 and Figure 3), this document organises the requirements regarding the implementation of ~~new~~ software into four subclauses:

- 6.6.1 provides requirements that are applicable whatever implementation technique is used;
- 6.6.2 provides requirements specific to the configuration of pre-developed software and of devices containing software, and in particular the setting of parameters and other configuration data;
- 6.6.3 provides requirements specific to the implementation and verification of software in application-oriented languages;
- 6.6.4 provides requirements specific to the implementation and verification of software in general-purpose languages.

As boxes titled “Application software development/generation” and “Development of ~~new~~ operational system software” represent a large and essential part of the software safety lifecycle, a zoom is provided in Figure 4 which illustrates in more detail the activities between software requirements specification and software validation, with a clear representation of the

three different implementation paths (configuration of pre-developed software and devices, use of application-oriented languages and use of general-purpose languages).

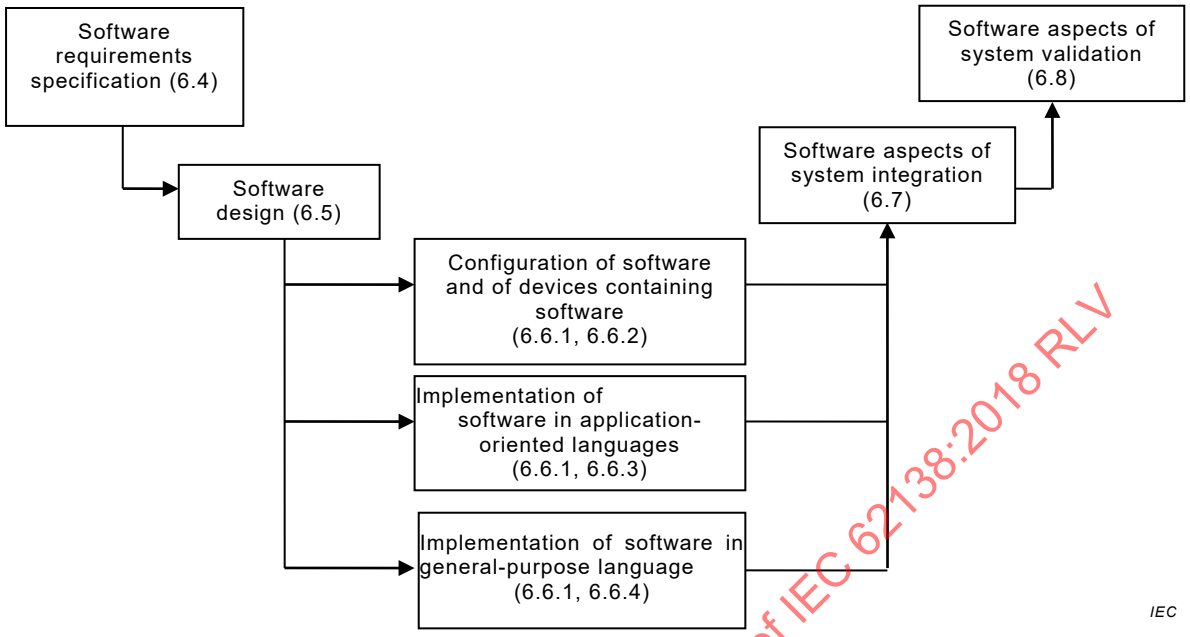
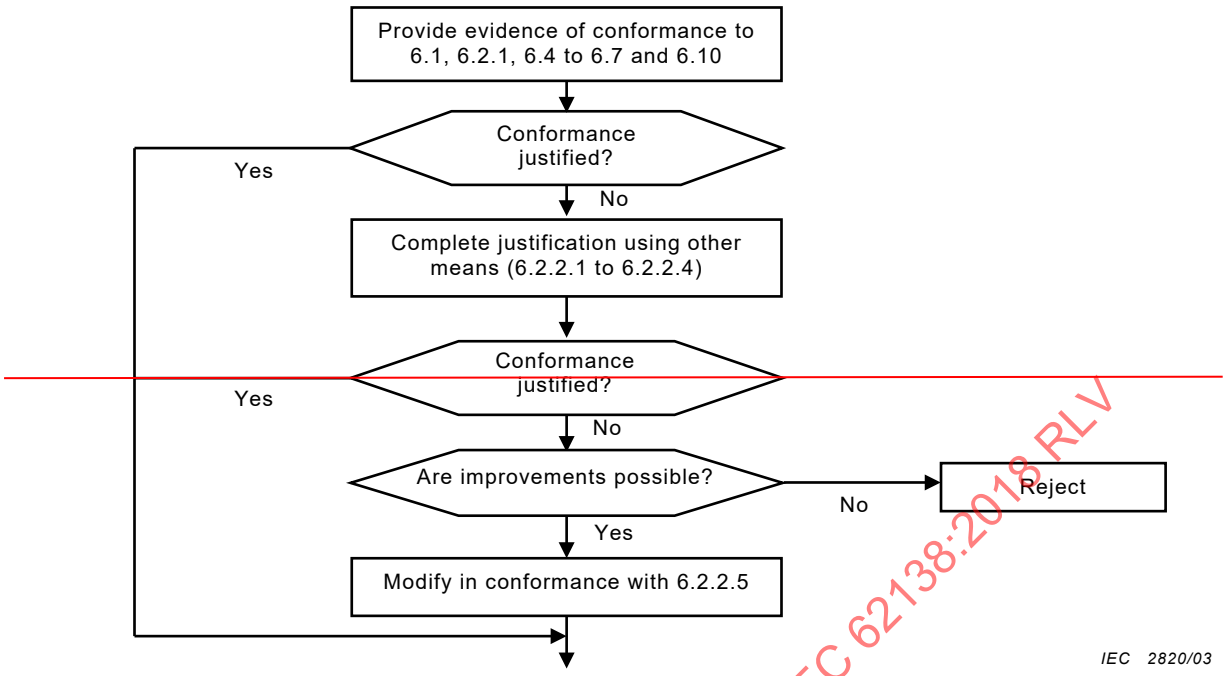


Figure 4 – Development activities of the IEC 62138 software safety lifecycle



~~Figure 5 – Process for providing evidence of correctness for pre-developed software of an I&C system of safety class 2~~

~~Another activity of particular importance in the Software Safety Lifecycle is the selection of pre-developed software, as this type of software usually represents a very significant portion of the final integrated software. Figure 5 illustrates in more detail the selection process to be used for safety class 2.~~

5.5 Gradation principles

As a consequence of the gradation of safety relevance for functions of categories A, B and C (see IEC 61226), a suitable gradation has been adopted for the requirements applicable to the software of I&C systems of safety classes 1, 2 and 3.

Software of I&C systems of safety classes 1 is covered by IEC 60880.

The application of the requirements of this document for safety class 3 confers the basic level of confidence that is suitable for software of an I&C system important to safety. The principles followed are:

– reliance on quality assurance;
– special attention given to the assurance that the software:
  - contributes as necessary to, and does not adversely affect, the functions important to safety;
  - satisfies the software requirements specification statements which define constraints important to safety;
– assurance that the operators of the I&C system are informed as early as reasonably possible of software errors and failures that may affect the functions identified as important to safety, so that any appropriate action can be taken;
– documented software requirements specifications, design specifications, integration specifications, validation specifications (i.e. full functional testing) and modification specifications.

For safety class 2, in addition to the principles already stated for class 3, the principles followed by this document are:

– justification, based on tests and design, that the required safety-related performance (for example, response times) will be met in all the specified conditions;

~~– use of Documentation for Safety for pre-developed software and for pre-developed devices with embedded software; the objective of such documentation is to provide all the information that is necessary for using the software or devices safely; in particular, the need for providing design-based justification regarding the safety-related performance sets the minimal level of information that is necessary;~~

~~– use of pre-developed software and devices with embedded software in accordance with rules based on the corresponding Documentations for Safety;~~

~~– configuration and use of pre-developed "black boxes" according to rules also aiming at the mitigation of the effects of the known or anticipated failure modes;~~

~~– justification of correctness and functional suitability for pre-developed software and pre-developed devices with embedded software; Figure 5 illustrates the process for providing such justification of correctness for safety class 2;~~

~~– extensive and documented verification of the detailed design and of the implementation of new software; this may include manual inspections, tool supported analysis and tests;~~

– more stringent requirements for the selection of pre-developed software;

– more stringent requirements for verification, configuration management, selection and use of software tools and languages~~, security and fault tolerance~~;

– explicit requirements for simplicity, clarity, precision, verifiability, testability and modifiability.

~~When the same requirement is applicable to both safety classes, the extent of the necessary justification of compliance may depend on the safety class. In particular, for class 3, this extent may be less thorough regarding the functions that are not identified as important to safety and that do not jeopardise the functions identified as important to safety.~~

~~Requirements for software of I&C systems of safety class 1 are to be found in IEC 60880 and IEC 60880-2.~~

~~The requirements and recommendations of this standard are stated in Clauses 5 (for class 3) and 6 (for class 2). These Clauses can be read and used independently from one another. They have the same structure, so that Subclauses 5.x.y and 6.x.y provide requirements and recommendations for the same topic. The requirements of Clause 6 that are not identical to those of Clause 5 (either because they were added or because they were modified) are written in italics. All the requirements and recommendations are indented and numbered, and in a given pair of Subclauses, those addressing the same issue have the same number. All other paragraphs are informative.~~

~~It is not the intention of this standard to prescribe a defined set of documentation, but rather to define the information which should be documented. The particular hierarchy and format of documentation adopted may vary, provided that the principles set out in this standard are addressed.~~

When requirements are applicable to both safety classes, the extent of the justification required to confirm compliance with this document may be moderated according to the safety class, i.e. for class 3, the extent of justification may be reduced compared to class 2. Also, the extend of justification for those functions which are 'not important to safety' in class 2 or 3 systems need only address how the design ensures that such functions do not jeopardise the functions which are identified as important to safety.

5 Requirements for the software of I&C systems performing category C functions[1]

5.1 General requirements

5.1.1 Software Safety Lifecycle – Software Quality Assurance

Subclause 6.2.1 of IEC 61513 provides requirements for Quality Assurance at the level of an I&C system. This Subclause provides additional requirements specific, or of particular importance, to software.

1 The development of software shall be performed according to a Software Safety Lifecycle. The provisions of this Software Safety Lifecycle shall be specified in a Quality Assurance Plan.

This Quality Assurance Plan may be a part of the System Quality Assurance Plan, or may be a separate Software Quality Assurance Plan.

2 If a separate Software Quality Assurance Plan is used, it shall be consistent with the System Quality Assurance Plan. The applicable requirements of 6.2.1 of IEC 61513 shall be addressed by the two plans.

3 The Quality Assurance Plan shall divide the development phase of the Software Safety Lifecycle into specified activities. These activities shall include the activities necessary to achieve the required software quality, and to verify and provide objective evidence that this quality is achieved.

4 The specification of an activity shall state:

- its objectives;
- its relationships and interactions with other activities;
- its inputs and results;
- the organisation and responsibilities relevant to the activity.

The contents and properties required of the inputs and results should also be specified.

5 The Quality Assurance Plan shall require that the implementation of each activity is assigned to competent persons equipped with adequate resources.

6 The Quality Assurance Plan shall require that modifications in already approved documents are identified, reviewed and approved by authorised persons.

7 The Quality Assurance Plan shall require that the methods, languages, tools, rules and standards used are identified and documented, and known to, and mastered by, the persons concerned.

8 The Quality Assurance Plan shall require that if several methods, languages, tools, rules and/or standards are used, it is clear which ones should be used for each activity.

9 The Quality Assurance Plan shall require that project-specific terms, expressions, abbreviations and conventions used are explicitly defined.

10 The Quality Assurance Plan shall require that the issues raised are tracked and resolved.

11 The Quality Assurance Plan shall require that records resulting from its application are produced. In particular, it shall require that the results of verifications and reviews are recorded together with the scope of the verifications or reviews, the conclusions reached and the resolutions agreed. Any deviation from the Quality Assurance Plan shall be documented and justified.

ISO 9000-3 provides additional guidelines for software Quality Assurance.

[1] The numbering of the items in the following Subclauses corresponds to that of Clause 6.

5.1.2 Verification

1 A Verification Plan shall define the scope of software verification and review activities.

2 Verifications and reviews shall be performed according to documented provisions. In particular, at stages of the Software Safety Lifecycle specified by the Verification Plan, the results of activities designated by the Verification Plan shall be verified to show that:

- the results are under configuration management;
- the activities have precisely identified inputs, and their results are consistent with these inputs;
- the activities fulfil their specified objectives, and their results have the required contents and properties, and comply with any resolution agreed;
- the results are clear, precise and up-to-date;
- the results comply with any applicable rule;
- the results comply with the applicable requirements of this standard.

"Precisely identified" means that the version is known without any ambiguity. "Clear" means that the individuals who need to read a document can fully understand it without excessive effort, even if they have not been involved earlier in the project, provided that they have the required knowledge. "Precise" means that there is no ambiguity.

The extent of the verification and review activities may be dependent on the scale and nature of the software, on the scale and nature of the results to be verified or reviewed, and on the methods and tools used. This extent may also be less thorough regarding the specified requirements that are not identified as important to safety (see requirement 6 of 5.3.3) and that cannot jeopardise the functions identified as important to safety.

3 The verification of the results of an activity shall be performed by competent persons who did not participate in the activity. This should include representatives of those concerned with the use of these results, as well as other experts, as necessary.

This does not imply that a person who is an author for one document cannot be the verifier of another.

4 The Software Requirements Specification, the Software Design Specification and the Software Validation Plan shall be verified.

5.1.3 Configuration management

Subclause 6.2.1.2 of IEC 61513 provides requirements for configuration management at the I&C system level. This Subclause provides additional requirements specific, or of particular importance, to software.

1 Configuration management for software shall be performed according to the provisions of a Configuration Management Plan or of the Quality Assurance Plan. These provisions shall be consistent with those for system level configuration management.

2 Configuration management shall be applied to the items related to the correctness of software. The Configuration Management Plan shall specify which software items or types of software items are to be held under configuration management. In particular, these shall include:

- the key documents of the Software Safety Lifecycle (in particular the documents required to be verified);
- the software components necessary to build the executable code, and the executable code itself;
- the software tools influencing the correctness of software and/or system design.

3 The Configuration Management Plan shall specify technical means for the authentication of the software items under configuration management and of their versions.

4 The Configuration Management Plan shall ensure that the version of the software attached to a given version of the system or equipment, and the versions of the items which together constitute this software version are uniquely identified.

5.1.4 Selection and use of software tools

Software tools can play an important role in preventing the introduction of faults in software or in system design, and in revealing already existing faults. In particular, tools can aid or automate the design of the architecture of I&C systems and the development of new application software.

1 Software tools should support the development activities which contribute to the correctness of software and system design.

It is usually preferable to focus not only on the quality and on the use of individual tools, but also to consider their compatibility with any other tools to be used, so that together, the tools selected form a coherent tool set. Generally it is preferable to use a well-known tool with extensive operational experience rather than an untried tool with no operational experience, but each case needs to be considered on its merits.

2 The equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in new application software.

These tools usually include support for application-oriented languages, allowing plant and system engineers to specify or verify applications functions. Other significant subjects for such tools may be animation, code generation and aid in the identification of functional test cases.

3 The equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in the configuration of their pre-developed software and in the design of the system.

Such tools may for example assist system designers in:

- organising the system into a suitable set of interconnected sub-systems;
- distributing the application functions across the sub-systems;
- configuring the sub-systems, their communications and their operational system software;
- ensuring that resources are adequate for all the modes of behaviour of the system;
- taking into account design and implementation constraints, in particular those aiming at the correctness and robustness of the system.

4 The Quality Assurance Plan shall precisely identify the software tools which may influence the correctness of software and/or system design.

5 User Documentation shall be provided for such tools to ensure that they are used as intended.

7 Evidence should be provided regarding the quality of the software tools which might introduce faults in software or in system design, and regarding their ability to produce correct results.

Code generators and compilers are examples of such tools.

Evidence regarding tool quality and ability to produce correct results may be based on operational experience, tool certification, certification of their suppliers for appropriate development practices, guarantee of appropriate tool development processes, and/or tests.

~~5.1.5 Selection of languages~~

~~1 The languages (application-oriented or general-purpose) used to develop software shall have precise and documented syntax and semantics.~~

~~2 Application-oriented languages, if available, should be preferred.~~

~~4 When more than one language is used for the same executable code, interfaces between languages shall be documented.~~

~~The interface between languages includes argument passing schemes and representation of data structures.~~

~~6 The general-purpose languages used should support static typing of variables.~~

~~Explicit and static typing of variables is recommended in preference to implicit or dynamic typing.~~

~~5.1.6 Security~~

~~The objective of security is to provide adequate confidence that unauthorised persons and systems can neither modify the software and its data nor gain access to the system functions, and yet to ensure that this is not denied to authorised persons and systems. Subclauses 5.4.2 and 6.2.2 of IEC 61513 provide requirements for security, at the level of the I&C architecture and of an individual I&C system. This Subclause provides additional requirements specific, or of particular importance, to software.~~

~~1 An analysis of the security threats and vulnerability regarding the software aspects of the I&C system shall be performed and documented. It should take into account the relevant phases of the System and Software Safety Lifecycles. It should determine the requirements regarding the protection, the accessibility, the confidentiality and the integrity of data and functions.~~

~~These may include:~~

- ~~identification of security critical data and functions;~~
- ~~identification and authentication of personnel;~~
- ~~access control to security critical data and functions;~~
- ~~management of security critical data and functions;~~
- ~~traceability of security related actions to personnel.~~

~~2 Software development shall be performed according to the provisions of a Security Assurance Plan or of the Quality Assurance Plan. These provisions shall take into account the results of the threat and vulnerability analysis. They shall be consistent with the requirements of 5.4.2 and 6.2.2 of IEC 61513.~~

~~3 When relevant, software should be configured and parameterised so as to avoid unnecessary causes of vulnerability.~~

~~4 The plan should include provisions for the evaluation of the effectiveness of the solutions implemented.~~

~~5.2 Selection of pre-developed software~~

~~Subclause 6.1.2.1 of IEC 61513 provides general requirements for the selection of pre-developed components (not necessarily software components). This Subclause provides additional requirements specific, or of particular importance, to software.~~

~~5.2.1 User documentation~~

~~5.2.1.1 Objectives~~

~~1 Pre-developed software shall have documentation giving the information necessary for using the software in the I&C system.~~

In this standard, the corresponding document or set of documents is called User Documentation. When the pre-developed software is a part of an equipment or equipment family, this documentation may be a part of the User Documentation of the equipment or equipment family.

5.2.1.2 Contents

1 User Documentation shall include a description of:

- the functions provided;
- the interfaces with applications;
- the roles, types, formats, ranges and constraints of inputs, outputs, exception signals, parameters and configuration data, where appropriate;
- the different modes of behaviour and the corresponding conditions of transition;
- any constraint to be respected when using the pre-developed software.

3 When applicable, the User Documentation should also provide information regarding the performances (for example, in terms of response time) of the functions.

Functions, interfaces and performances may depend on the mode of behaviour, on the values of the parameters, on the configuration data and on the conditions provided to the software.

5.2.1.3 Properties

1 User Documentation shall be precise so as to avoid divergent interpretations.

5.2.2 Evidence of correctness

5.2.2.1 General requirements

1 The correctness of pre-developed software with respect to its User Documentation shall be justified.

The justification is usually qualitative because there are no generally recognised means to quantify it. In the case of pre-developed software, it may be based on different types of evidence, for example:

- evidence of conformance to all or part of the requirements of 5.1, 5.2.1, 5.4, 5.5, 5.6, 5.7 and 5.10 when applicable;
- project specific complementary tests;
- applicable and credible operational experience;
- relevant certification by credible authorities.

Confidence may be easier to obtain when the pre-developed software can be used only in a limited number of different ways, and/or when the design of the I&C system and of its software guarantees a well-defined set of conditions of use.

5.2.2.2 Complementary tests

No requirement is necessary for class 3 regarding this subject.

5.2.2.3 Operational experience

No requirement is necessary for class 3 regarding this subject.

5.2.2.4 Certification

No requirement is necessary for class 3 regarding this subject.

~~**5.2.2.5 Modification**~~

~~No requirement is necessary for class 3 regarding this subject.~~

~~**5.2.3 Functional suitability**~~

~~No requirement is necessary for class 3 regarding this subject.~~

~~**5.2.4 Selection and use of dedicated devices with embedded software**~~

~~Black-box devices with embedded software may be used in an I&C system under the following conditions.~~

~~1 The software of the device shall be integrated in such a way that it cannot be modified by the user and cannot be used separately from the rest of the device.~~

~~2 The number of the functions of the device, its potential for configuration, and the extent of its interfaces and interactions with the rest of the I&C system shall be limited so as to allow a thorough functional coverage by tests.~~

~~3 Evidence shall be given that the device itself conforms to the requirements of 5.2.1 and 5.2.2.~~

~~4 Evidence shall be given that the configuration and the use of the device in the I&C system conforms to the requirements of 5.4, 5.5.1 and 5.5.2.~~

~~**5.3 Software requirements specification**~~

~~This Subclause completes and adds precision to the requirements of 6.1.2.3 of IEC 61513.~~

~~**5.3.1 Objectives**~~

~~1 The requirements for the software of an I&C system shall be specified and documented.~~

~~In this standard, the corresponding document or set of documents is called the Software Requirements Specification. In principle, its objective is to specify what the software is to achieve without specifying how it must do it. However, design and implementation constraints may have to be specified when this is required by considerations of the design of the I&C system or of the I&C architecture.~~

~~3 The Software Requirements Specification shall be such that:~~

~~– it contributes to the confidence in the correctness of the design of the I&C system;~~

~~– compliance of the I&C system to the requirements of IEC 61513 can be demonstrated.~~

~~The IEC 61513 requirements concerned with Software Requirements Specification are mainly in 6.1.1.2, 6.1.1.3, 6.1.1.4, 6.1.2.2, 6.1.2.4 and 6.1.3.~~

~~4 The Software Requirements Specification shall be a reference for software design, software validation, and possible software modifications.~~

~~**5.3.2 Inputs**~~

~~3 The references, if any, made by the Software Requirements Specification to other documents shall be precise so as to be unambiguous.~~

~~**5.3.3 Contents**~~

~~1 The Software Requirements Specification shall specify:~~

~~– the application functions to be provided by the software;~~

- the different modes of behaviour of the software, and the corresponding conditions of transition;
- the interfaces and interactions of the software with its environment (for example, with operators, with the rest of the I&C system, with the other systems and equipment with which it interacts or shares resources), including the roles, types, formats, ranges and constraints of inputs and outputs;
- the parameters of the software which are to be modified by operators during operation, if any, their roles, types, formats, ranges and constraints, and the checks to be performed by the software when they are modified;
- required performance, when appropriate;
- what the software must not do or must avoid, when appropriate;
- the requirements of, or the assumptions to be made by, the software regarding its environment, when applicable.

2 The Software Requirements Specification should also specify the conditions (for example, the demand load), in particular the worst case conditions, provided to the software by its environment.

Functions, interfaces and performances requirements may depend on the mode of behaviour, on the values of the parameters, on the configuration data and on the conditions provided to the software.

3 The Software Requirements Specification shall specify the software modes of behaviour required when errors or failures are detected. When periodic tests are required of the I&C system, the Software Requirements Specification shall also specify the mode of behaviour required when such tests are performed.

4 The Software Requirements Specification should state the software quality objectives and shall state the constraints to be respected by software design and implementation for the sake of correctness and robustness.

For example, this may include constraints:

- to give confidence in the correctness of software and system design (for example, margins to be taken when using dynamically allocated resources such as memory, processing power, communication bandwidth, operating system resources);
- to enhance the ability of the software and of the I&C system to tolerate faults, to detect and signal errors and failures, to take specified modes of behaviour and to recover from failures;
- to give confidence that mistakes of operators and failures of other systems or equipment with which the software interacts or shares resources will not lead to unacceptable effects.

5 The Software Requirements Specification shall specify the contribution of the software to the assurance that the operators will be informed in due time of errors or failures concerning the functions of the I&C system identified as important to safety. The information provided to the operators shall allow them to take any appropriate action.

6 The Software Requirements Specification shall identify the functions and the requirements related to safety category C.

5.3.4 Properties

No requirement is necessary for class 3 regarding this subject.

5.4 Software design

5.4.1 Objectives

1 The design of software shall be documented. The documentation should give an overview of the organisation and of the functioning of the software.

In this standard, the corresponding document or set of documents is called the Software Design Specification. When pre-developed software is used, the Software Design Specification may reference the corresponding User Documentation.

3 The Software Design Specification shall provide evidence that the Software Requirements Specification statements important to safety are taken into account and will be satisfied in all specified conditions.

5 The Software Design Specification shall ensure, if applicable, that the adverse side effects of software errors and failures are cleared prior to returning to a normal mode of behaviour.

7 The Software Design Specification shall be a reference for software implementation and integration, and for possible software modifications.

5.4.2 Inputs

1 The inputs to the software design process shall include the Software Requirements Specification and the User Documentation of pre-developed software.

They may also include other documents, such as project specific constraints, and/or applicable rules and standards.

5.4.3 Contents

1 The Software Design Specification shall include the specification of:

- the overall organisation of the software;
- the overall functioning of the software under the conditions and modes of behaviour required by Software Requirements Specification.

2 The overall organisation should provide information regarding:

- the precise identification and the configuration of pre-developed software;
- the distribution of resources, software components and software tasks over sub-systems;
- the allocation of software (sub-)functions and performances to the identified software tasks;
- the main internal interfaces, in particular the interfaces between software tasks.

The overall functioning should provide information regarding:

- interactions, communication protocols and information flows;
- sequencing and timing constraints;
- use of resources;
- synchronisation, particularly when using shared resources.

5.4.4 Properties

No requirement is necessary for class 3 regarding this subject.

5.5 Implementation of new software

5.5.1 General requirements

The requirements of this Subclause are applicable to all new software, i.e. to the configuration of pre-developed software, and to programs written in application-oriented or general-purpose languages.

1 The use of pre-developed software shall be verified to be consistent with the corresponding User Documentation and with the constraints set by the Software Design Specification.

2 The procedures used to translate new programs into executable code shall be documented and verified.

5.5.2 Configuration of software and of devices containing software

The requirement of this Subclause is specific to the configuration of customisable software. Such software may be pre-developed or new. The requirement is also applicable to the configuration of customisable devices with embedded software. However, when the configuration data represents processing to be performed by the software or the system (i.e. it is effectively software programs), 5.5.3 applies.

1 The configuration of customisable software and devices with embedded software shall be documented.

5.5.3 Implementation with application-oriented languages

The requirements of this Subclause are specific to programs written in application-oriented languages. Generally, application oriented formats (such as logic diagrams or function block diagrams) may be used to express all or part of the Software Requirements Specification or of the Software Design Specification. Only limited detailed design and implementation effort is then necessary to transform the specification into programs that can be automatically translated into executable code.

1 The parts of the Software Requirements Specification and/or of the Software Design Specification that are used to generate executable code by automated means shall be considered to be programs written in application-oriented languages.

2 Programs written in application-oriented languages which are related to functions important to safety shall be verified to be functionally correct and consistent.

5.5.4 Implementation with general-purpose languages

The requirement of this Subclause is specific to programs written in general-purpose languages.

4 Programs written in general-purpose languages shall conform to documented rules aiming at clarity, modifiability and testability.

A set of rules may be specific to a language or to a set of programs.

5.6 Software aspects of system integration

The integration of software is considered as part of system integration. This Subclause complements 6.1.4 and 6.2.3 of IEC 61513 by providing additional requirements specific, or of particular importance, to software.

1 Software integration and/or inspections shall show that the integrated system and the software:

- comply with the design provisions that ensure the satisfaction of the Software Requirements Specification statements identified as important to safety;
- satisfy the constraints stated by the Software Requirements Specification with respect to correctness and robustness.

3 Software integration shall be performed according to the provisions of the System Integration Plan or of a Software Integration Plan.

4 Records of the application of the plan used for software integration shall be produced, for example, test results. In the event of software or system modifications being required, it

shall be possible to repeat all, or a subset of, the integration tests to evaluate the extent of possible changes in behaviour.

5.7 Software aspects of system validation

The objective of the validation of software is to ensure compliance of the integrated software with the functional, performance and interface specifications imposed by the I&C system requirements. Thus, it is considered as part of system validation. This Subclause complements 6.1.5 and 6.2.4 of IEC 61513 by providing additional requirements specific, or of particular importance, to software.

1 Software validation shall show that, in the target I&C system, the integrated software conforms to the functional, performance and interface requirements that are identified as important to safety. This shall include justification that:

- the specified software functions important to safety are correctly performed when their parameters and inputs are in the ranges specified by the Software Requirements Specification, in the conditions of use defined in the Software Requirements Specification;
- the system functions important to safety to which the software contributes are correctly performed in the conditions of use defined in the System Requirements Specification;
- the software provides defences as required by the Software Requirements Specification against errors of operators and failures of other systems and equipment;
- the plant engineering data used by, or integrated in, the I&C system to implement functions important to safety is correct; in particular, the validation of the software shall show that this data correctly describes and addresses the systems and equipment of the plant with which the software interacts or shares resources.

NOTE For some aspects of validation testing, it may be acceptable to use a hardware platform identical to the actual final target platform if adequate justification is provided.

The conditions of use of functions important to safety may include the concurrent operation of functions not important to safety.

2 Software validation should be performed according to the provisions of the System Validation Plan. If not, it shall be performed according to the provisions of a Software Validation Plan.

3 The plan used for software validation shall specify the validation actions to be performed, and shall show that all the functionality, performance and interface statements of the Software Requirements Specification identified as important to safety are correctly taken into account by these actions. It shall also specify the main phases of the software validation (for example, an off-site phase followed by an on-site phase) and the corresponding means, methods and tools to be used.

4 Records of the application of the plan used for software validation shall be produced. In the event of software or system modifications being required, it shall be possible to repeat all, or a subset of, the validation tests to evaluate the extent of possible changes in behaviour. The results of software validation should be auditable by persons competent in the subjects addressed but not directly engaged in the validation process.

5 These records shall document the configuration of the software being validated and the configuration of the validation environment (for example, the hardware environment and the tools, if any).

6 The team that writes the plan used for software validation shall include at least one person who did not participate in the design and implementation.

5.8 Installation of software on site

Subclause 6.1.6 of IEC 61513 provides requirements regarding the installation of the I&C system on site. This Subclause provides additional requirements specific, or of particular importance, to the installation of software.

1 The procedure for installing software on site shall be documented. It shall guarantee that the correct and complete version of the software is installed.

2 The procedure for installing software on site shall include and specify on-site checks and tests to be performed before the I&C system is put into full operational use. In particular, the satisfaction of the conditions required for correct operation of the software shall be verified.

For example, these conditions may concern the hardware on which the software operates, or other systems with which the software interacts or shares resources.

5.9 Anomaly reports

1 If unexpected, apparently incorrect, unexplained or abnormal behaviour is experienced after acceptance into service, an anomaly report should be raised.

2 The anomaly report should give details of the behaviour, the software and hardware configurations and the activities in hand at the time. It should also include the originator, location, date, and a report identification.

The report identification may be added following an initial review of the report to ensure that it is valid.

3 The anomaly reports should be reviewed. Issues raised should be documented, tracked and resolved.

5.10 Software modification

The decision to proceed with software modifications depends upon their impact on the I&C system. Therefore, they are subject to the requirements of 6.1.7 and 6.3.6 of IEC 61513. This Subclause provides additional requirements specific, or of particular importance, to software.

1 Software modifications shall be developed so as to maintain consistency with the requirements of 5.1, 5.2, 5.3, 5.4 and 5.5. They shall be installed on-site in accordance with the requirements of 5.8.

2 Software modifications should be integrated and validated in a manner consistent with 5.6 and 5.7. When the extent of the modification does not necessitate a full application of these two Subclauses, the integration of the modified software shall be performed according to a Regression Software Integration Plan, and the validation shall be performed according to a Regression Software Validation Plan. The adequacy and thoroughness of these plans shall be justified taking into account the extent of any modifications made in the Software Requirements Specification and in the Software Design Specification. Records of the application of these plans shall be produced.

4 Software modifications shall be comprehensively documented. In particular, all affected software documents shall be updated.

7 The effects of a software modification on the rest of the I&C system and on the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

8 The effects on software of modifications in the rest of the I&C system or in the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

6 Requirements for the software of class 2 and class 3 I&C systems ~~performing category B functions~~

~~Clause 6 provides additional requirements for software of I&C systems performing category B functions (i.e., systems of safety class 2); in order to facilitate the use of the standard, this clause repeats the relevant requirements for class 3; the additional or modified requirements are in italics.~~

6.1 Applicability of the requirements

The requirements and recommendations of this document are stated in this Clause 6. The requirements and recommendations that are not specifically marked are applicable to class 2 and class 3 systems. The requirements and recommendations that are applicable specifically to class 3 or to class 2 systems are identified as such and appear in italics.

All the requirements and recommendations are indented and numbered. All other paragraphs are informative. In particular, unnumbered paragraphs provide notes regarding the immediately preceding numbered paragraph unless otherwise stated. When unnumbered paragraphs provide notes regarding more clauses than the immediately preceding numbered paragraph, they are introduced by “Concerning xxx and yyy, …”.

It is not the intention of this document to prescribe a defined set of documents, but rather to define the information which needs to be documented. The particular hierarchy and format of documentation adopted may vary, provided that the principles set out in this document are addressed. For information Annex A presents a typical list of software documentation.

6.2 General requirements

6.2.1 Software safety lifecycle – Software quality assurance

6.2.1.1 General

Subclause 6.3.2 of IEC 61513:2011 provides requirements for quality assurance at the level of an I&C system. This subclause provides additional requirements specific, or of particular importance, to software.

6.2.1.2 The development of software shall be performed according to a software safety lifecycle. The provisions of this software safety lifecycle shall be specified in a quality assurance plan.

This quality assurance plan may be a part of the system quality assurance plan, or may be a separate software quality assurance plan.

6.2.1.3 If a separate software quality assurance plan is used, it shall be consistent with the system quality assurance plan. ~~The applicable requirements of 6.3.2 of IEC 61513 shall be addressed by the two plans.~~ The software quality assurance plan shall address the requirements of 6.3.2 of IEC 61513:2011 as they relate to software.

6.2.1.4 The quality assurance plan shall divide the development phase of the software safety lifecycle into specified activities. These activities shall include the activities necessary to achieve the required software quality, and to verify and provide objective evidence that this quality is achieved.

6.2.1.5 The specification of an activity shall state:

- its objectives;
- its relationships and interactions with other activities;

- its inputs and results;
- the organisation and responsibilities relevant to the activity.

6.2.1.6 The contents and properties required of the inputs and results should also be specified.

6.2.1.7 The quality assurance plan shall require that the implementation of ~~these activities~~ each activity is assigned to competent persons equipped with adequate resources.

6.2.1.8 The quality assurance plan shall require that modifications in ~~already~~ approved documents are identified, reviewed and approved by authorised persons.

6.2.1.9 The quality assurance plan shall require that the methods, languages, tools, rules and standards used are identified and documented, ~~and~~ known to, and ~~mastered by,~~ within the ~~persons~~ competencies of the concerned development personnel.

6.2.1.10 The quality assurance plan shall require that if several methods, languages, tools, rules and/or standards are used, it is clear which ones ~~should~~ have to be used for each activity.

6.2.1.11 The quality assurance plan shall require that project specific terms, expressions, abbreviations and conventions used are explicitly defined.

6.2.1.12 The quality assurance plan shall require that ~~the issues~~ non-conformances raised are tracked and resolved.

6.2.1.13 The quality assurance plan shall require that records resulting from its application are produced. In particular, it shall require that the results of verifications and reviews are recorded together with the scope of the verifications or reviews, the conclusions reached and the resolutions agreed. Any deviation from the quality assurance plan shall be documented and justified.

~~ISO 9000-3 provides additional guidelines for software Quality Assurance.~~

6.2.1.14 The quality assurance plan shall require that the output documentation constitutes a set of appropriately cross-referenced mutually consistent documents, ensuring the traceability of the final design to the input requirements.

6.2.2 Verification

6.2.2.1 A verification plan shall define the scope of software verification and review activities.

6.2.2.2 The verification plan shall address the requirements of 6.3.2.2 of IEC 61513:2011 as they relate to software.

6.2.2.3 Verifications and reviews shall be performed according to documented provisions. ~~In particular, at stages of the Software Safety Lifecycle specified by the Verification Plan, the results of activities designated by~~ The Verification Plan shall ~~be verified to show~~ ensure that:

- the verification results are held under configuration management;
- all verification activities have precisely identified inputs, and their results are consistent with these inputs;
- the activities fulfil their specified objectives, and their results have the required contents and properties, and comply with any resolution agreed;
- the results are clear, precise and up-to-date;

- the results comply with any applicable rule;
- the results comply with the applicable requirements of this document.

“Precisely identified” means that the version is known without any ambiguity. “Clear” means that the individuals who need to read a document can fully understand it without excessive effort, even if they have not been involved earlier in the project, provided that they have the required knowledge. “Precise” means that there is no ambiguity.

The extent of the verification and review activities may be dependent on the scale and nature of the software, on the scale and nature of the results to be verified or reviewed, and on the methods and tools used. This extent may also be less thorough regarding the specified requirements that are not identified as important to safety (see 6.4.4.7) and that cannot jeopardise the functions identified as important to safety.

6.2.2.4 The verification plan should ensure that records are produced such that the verification process is fully auditable, i.e. such that independent confirmation of the implementation of the verification plan may be performed.

6.2.2.5 The verification of the results of an activity shall be performed by competent persons who did not participate in the activity.

This does not imply that a person who is an author for one document cannot be the verifier of another.

6.2.2.6 ~~This~~ The verification of the results of an activity should include representatives of those concerned with the use of these results, as well as other experts, as necessary.

6.2.2.7 The software requirements specification, the software design specification and the software validation plan shall be verified.

6.2.2.8 *For class 2, the application of design and implementation rules shall be verified.*

6.2.2.9 Software verification shall be performed by persons who did not develop the software being verified.

6.2.2.10 *For class 2, persons who do the verification should have managerial independence from the developers.*

6.2.3 Configuration management

6.2.3.1 General

Subclause 6.3.2.3 of IEC 61513:2011 provides requirements for configuration management at the I&C system level. This subclause provides additional requirements specific, or of particular importance, to software.

6.2.3.2 Configuration management for software shall be performed according to the provisions of a configuration management plan or of the quality assurance plan. These provisions shall be consistent with those for system level configuration management.

6.2.3.3 Configuration management shall be applied to the items related to the correctness of software. The configuration management plan shall specify which software items or types of software items are to be held under configuration management. In particular, these shall include:

- the key documents of the software safety lifecycle (in particular the documents required to be verified);

- the software components necessary to build the executable code, and the executable code itself;
- the software tools influencing the correctness of software ~~and/or system design~~.

6.2.3.4 The configuration management plan shall specify technical means for the authentication of the software items under configuration management and of their versions.

6.2.3.5 The configuration management plan shall ensure that the version of the software attached to a given version of the system or equipment, and the versions of the items which together constitute this software version are uniquely identified.

6.2.4 Selection and use of software tools

6.2.4.1 General

Software tools can play an important role in preventing the introduction of faults in software ~~or in system design,~~ and in revealing existing faults. In particular, tools can aid or automate the design of the architecture of I&C systems and the development of new application software.

6.2.4.2 Software tools should support the development activities which contribute to the correctness of software ~~and system design~~.

It is usually preferable to focus not only on the quality and on the use of individual tools, but also to consider their compatibility with any other tools to be used, so that together, the tools selected form a coherent tool set. Generally it is preferable to use ~~a well-known~~ tools with extensive and relevant operational experience ~~rather than an untried tool with no operational experience, but each case needs to be considered on its merits~~. The use of other tools may be justifiable based upon the requirements of a particular development process.

6.2.4.3 *For class 2, the equipment families used for the development of an I&C system shall be associated with software tools that can reduce the risk of introducing faults in new application software*.

6.2.4.4 *For class 3, the equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in new application software.*

Concerning 6.2.4.3 and 6.2.4.4, these tools usually include support for application-oriented languages, allowing plant and system engineers to specify or verify application functions. Other significant ~~subjects for~~ features of such tools may ~~be~~ include functional animation, automatic code generation and ~~aid~~ assistance in the ~~identification~~ development of functional test ~~cases~~ specifications.

6.2.4.5 The equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in the configuration of their pre-developed software and in the design of the system.

Such tools may for example assist system designers in:

- organising the system into a suitable set of interconnected sub-systems;
- distributing the application functions across the sub-systems;
- configuring the sub-systems, their communications and their operational system software;
- ensuring that resources are adequate for all the modes of ~~behaviour~~ operation of the system;
- taking into account design and implementation constraints, in particular those aiming at the correctness and robustness of the system.

6.2.4.6 The quality assurance plan shall precisely identify the software tools which may influence the correctness of software ~~and/or system design~~.

6.2.4.7 User documentation shall be provided for such tools to ensure that they are used as intended.

6.2.4.8 The quality assurance plan shall distinguish the tools which might introduce faults in software ~~or in system design,~~ from those which might only lead to overlooking already existing faults.

Code generators and compilers are examples of tools of the first category, whereas static code analysers and test case generators are examples of tools of the second category.

6.2.4.9 *For class 2, the software tools which might introduce faults in software ~~or in system design~~ shall be selected and used according to documented procedures and rules aiming at reducing or mitigating this risk. Evidence shall be provided regarding their quality and their ability to produce correct results. ~~Their use shall be traced so that the tools, if any, which were used to generate a given item or information may be identified.~~ Where tools have been applied to generate a given item or information their use shall be recorded to identify them.*

6.2.4.10 *For class 3, evidence should be provided regarding the quality of the software tools which might introduce faults in software and regarding their ability to produce correct results.*

6.2.4.11 Evidence regarding tool quality and ability to produce correct results ~~may~~ should be based on operational experience, tool qualification or certification, certification of their suppliers for appropriate development practices, guarantee of appropriate tool development processes, and/or tests. The required stringency of the evidence ~~may depend on~~ should be determined based upon the conditions of use of the tool, the extent of the verification of its outputs, the likelihood of tool errors to be detected, and the seriousness of the consequences of undetected erroneous results. Conversely, stringent evidence (for example, a tool qualification according to IEC 60880 ~~-2~~) may be used as a substitute for some of the verifications of outputs.

6.2.4.12 *For class 2, the software tools which might ~~lead~~ fail to report faults ~~already existing~~ in software ~~or in system design being overlooked~~ should be selected and used in a way which reduces this risk. ~~Their use should be traced.~~*

6.2.4.13 *For class 2, the use of software tools which might fail to report faults in software should be recorded.*

6.2.4.14 *For class 2, when a tool or tool version which has the potential to introduce faults in software ~~or in system design~~ is substituted with another, ~~reasonable~~ precautions shall be taken to ensure that this does not have adverse effects on the correctness of the software ~~and the system design~~.*

For example, in addition to the quality and ability of the new tool to produce correct results, its compatibility with the previous tool may need to be assessed.

6.2.5 Selection of languages

6.2.5.1 The languages (application-oriented or general-purpose) used to develop software shall have precise and documented syntax and semantics.

6.2.5.2 Application-oriented languages, if available, should be ~~preferred~~ used.

6.2.5.3 *For class 2, low level, machine-oriented general-purpose languages (for example, assembly languages) may be used for specific ~~software~~ computer programs, but this should be justified.*

6.2.5.4 When more than one language is used for ~~the same~~ generating executable code, interfaces between languages shall be documented.

The interface between languages includes argument passing schemes and representation of data structures.

6.2.5.5 *For class 2, the general-purpose languages used should have features facilitating tool supported static analyses of computer programs.*

6.2.5.6 ~~In particular,~~ The general-purpose languages used should support explicit and static typing of variables.

6.2.5.7 *For class 2, explicit and static typing of variables should be used ~~in preference to implicit or dynamic typing~~.*

6.2.5.8 *For class 2, the languages used and their corresponding run-time libraries shall enable predictable run-time behaviour of the software.*

For example, disruption of the normal behaviour of the software for the collection of freed memory at random moments is usually not acceptable.

~~**6.1.6 Security**~~

~~The objective of security is to provide adequate confidence that unauthorised persons and systems can neither modify the software and its data nor gain access to the system functions, and yet to ensure that this is not denied to authorised persons and systems. Subclauses 5.4.2 and 6.2.2 of IEC 61513 provide requirements for security, at the level of the I&C architecture and of an individual I&C system. This Subclause provides additional requirements specific, or of particular importance, to software.~~

~~1 An analysis of the security threats and vulnerability regarding the software aspects of the I&C system shall be performed and documented. *It shall take into account the relevant phases of the System and Software Safety Lifecycles. It shall determine the requirements regarding the protection, the accessibility, the confidentiality and the integrity of data and functions.*~~

~~These may include:~~

- ~~identification of security critical data and functions;~~
- ~~identification and authentication of personnel;~~
- ~~access control to security critical data and functions;~~
- ~~management of security critical data and functions;~~
- ~~traceability of security related actions to personnel.~~

~~2 Software development shall be performed according to the provisions of a Security Assurance Plan or of the Quality Assurance Plan. These provisions shall take into account the results of the threats and vulnerability analysis. They shall be consistent with the requirements of 5.4.2 and 6.2.2 of IEC 61513.~~

~~3 When relevant, software should be configured and parameterised so as to avoid unnecessary causes of vulnerability.~~

~~*4 The Plan shall include provisions for the evaluation of the effectiveness of the solutions implemented.*~~

6.3 Selection of pre-developed software

6.3.1 General

Subclause 6.2.3.2 of IEC 61513:2011 provides general requirements for the selection of pre-~~developed~~ existing components (not necessarily software components). This Subclause 6.3 provides additional requirements specific, or of particular importance, to software.

6.3.1.1 Application software is plant specific and so should not be considered pre-developed software.

NOTE The same application software may be used in multiple units based on the same plant design and safety requirements. In such a case, the justifications produced for the original unit are applicable for the following units.

6.3.2 Documentation for safety

6.3.2.1 Objectives

6.3.2.1.1 Pre-developed software shall have documentation giving the information necessary for using the software safely in the I&C system ~~and for providing evidence that the requirements of 6.2.3 (Functional suitability) and 6.4 (Software design) are satisfied~~.

In this document, the corresponding document or set of documents is called documentation for safety. When the pre-developed software is a part of an equipment or equipment family, this documentation may be a part of the documentation for safety of the equipment or equipment family.

Documentation for safety ~~may be generic or project specific. It may comprise~~ generally comprises more than the user documentation provided by the supplier of the pre-developed software. For example, it may include information obtained from additional tests, measurements and/or analyses, and from operational experience.

6.3.2.2 Contents

6.3.2.2.1 Documentation for safety shall include a description of:

- the functions provided;
- the interfaces with application~~s~~ software;
- the roles, types, formats, ranges and constraints of inputs, outputs, exception signals, parameters and configuration data, where appropriate;
- the different modes of ~~behaviour~~ operation and the corresponding conditions of transition;
- any constraint to be respected when using the pre-developed software.

6.3.2.2.2 *For class 2, when applicable, these constraints should:*

- *give adequate confidence in the correctness of the integrated software and of the system design (for example, margins to be taken when using dynamically allocated resources such as memory, processing power, communication bandwidth, operating system resources);*
- *enhance the ability of the integrated software and of the I&C system to detect, signal and tolerate failures, to adopt specified modes of ~~behaviour~~ operation and to recover from failures;*
- *give adequate confidence that operator mistakes and failures of other systems or equipment with which the integrated software interacts or shares resources will lead to defined modes of ~~behaviour~~ operation;*
- *guarantee that the environment of the pre-developed software will provide all the necessary resources in all conditions of use in the I&C system.*

6.3.2.2.3 When applicable, the documentation for safety should also provide information regarding the performance (for example, in terms of response time) of the functions.

~~Functions, interfaces and performances may depend on the mode of behaviour, on the values of the parameters, on the configuration data and on the conditions provided to the software.~~

The functions provided by the software, including those related to the system interfaces, may vary depending on the operational conditions of the plant.

6.3.2.2.4 *For class 2, the documentation for safety shall also provide information regarding:*

- *the self-~~surveillance~~ supervision performed, the fault tolerance capability and the failure modes;*
- *the requirements of the pre-developed software regarding its runtime environment (for example, regarding hardware or other software components);*
- *the interactions and interfaces of the pre-developed software with the hardware, to the extent necessary to fully define the safe functional performance of the system.*

6.3.2.2.5 *For class 2, the documentation for safety of the operational system software of a pre-developed equipment family shall provide information enabling (when combined with application-specific data) correct predictions regarding the key safety significant elements of system performance, ~~for example,~~ including notably the maximum response times ~~of, or~~ and the maximum usage of resources ~~by, compliant applications~~.*

Such information may be provided in the form of data, formulae and/or models allowing the calculation of worst case response times and the resource~~s~~ usage of applications. When the software offers a wide range of functions, interfaces and possibilities for configuration, an appropriate confidence in the correctness of the information may be difficult to obtain without knowledge of the ~~functioning~~ operating principles of the software.

6.3.2.3 Properties

6.3.2.3.1 Documentation for safety shall be ~~precise so as to~~ accurate and shall avoid ~~divergent interpretations~~ ambiguity.

6.3.3 Evidence of correctness

6.3.3.1 General requirements

6.3.3.1.1 The correctness of pre-developed software with respect to its documentation for safety shall be justified.

The justification is usually qualitative because there are no generally recognised means to quantify it. Figure 5 and Figure 6 ~~illustrates the approaches~~ describe a typical process that can be taken.

- ~~*when pre-developed software has been developed in conformance to the applicable requirements of 6.1, 6.2.1, 6.4, 6.5, 6.6, 6.7 and 6.10 when applicable, no additional justification of correctness is necessary;*~~
- ~~*when conformance to these requirements cannot be adequately justified, then 6.2.2.2, 6.2.2.3 and 6.2.2.4 provide complementary means which may be used to complete the justification.*~~

It is recognized however that this is not the only possible approach and that other approaches may be used.

6.3.3.1.2 When using complementary means for providing evidence of correctness, the acceptance criteria should be specified and justified in early stages of the software safety lifecycle. These criteria should be justified considering the requirements of this document the compliance to which has not been adequately established.

~~Confidence may be easier to obtain when a pre-developed software can be used only in a limited number of different ways, and/or when the design of the I&C system and of its software guarantees clearly defined conditions of use.~~

6.3.3.1.3 Pre-developed software should be divided into two different types:

a) Complete operational system software.

b) Software components (real time operating system, library, firmware).

NOTE Application software is plant specific, so it is not considered pre-developed software (see 6.3.1.1).

The rationale behind this distinction is that software components need to be integrated into larger software to form complete operational system software. This means that software components benefit from the development process of the complete operational system software where they are integrated. This allows their functionalities to be verified and validated in the context of their use in the complete operational system software. Therefore the recommended approach to justify the correctness of complete operational system software with respect to its Documentation for Safety (see Figure 5) is more demanding than the recommended approach to justify the correctness of software components (see Figure 6).

6.3.3.1.4 A pre-developed software item should be considered a software component only if it is integrated into larger software to form complete operational system software. Also, a pre-developed software item should be considered a software component only if a later reconfiguration of operational software could not lead to a component being executed in a different way to its initial use (as this would mean that the qualification performed on the complete operational unit would not adequately qualify the software component).

6.3.3.1.5 Verification and validation of the complete operational system software should be performed with the software components embedded. The integration may be within software that runs on a single processor, for example for real time operating systems or libraries. The integration may also be within software that run in close cooperation on several processors, for example the firmware of communication modules or input/output modules.

6.3.3.1.6 *For class 2, the qualification process for software components (see Figure 6) should be used only for pre-developed software components that are non-autonomous executables.*

An 'autonomous executable' is software that can be run by itself without any additional code.

General purpose operating systems designed primarily to be used on workstations are typically autonomous executables in the sense that once they are installed they automatically run many tasks which are not defined by the user.

Libraries (e.g. C library) need to be called by additional code in order to function. A library can be compiled and loaded onto a processor but it will not run unless code has been written to call the functions of the library. A library is therefore not an autonomous executable.

Real time operating systems designed primarily to run embedded software are usually non-autonomous executables in the sense that the user has to define each task explicitly, but this has to be checked on a case by case basis.

6.3.3.1.7 The correctness of a software component with respect to its documentation for safety should be justified by relevant, sufficient and positive operational experience (see 6.3.3.3) or by certification (see 6.3.3.4) (see Figure 6).

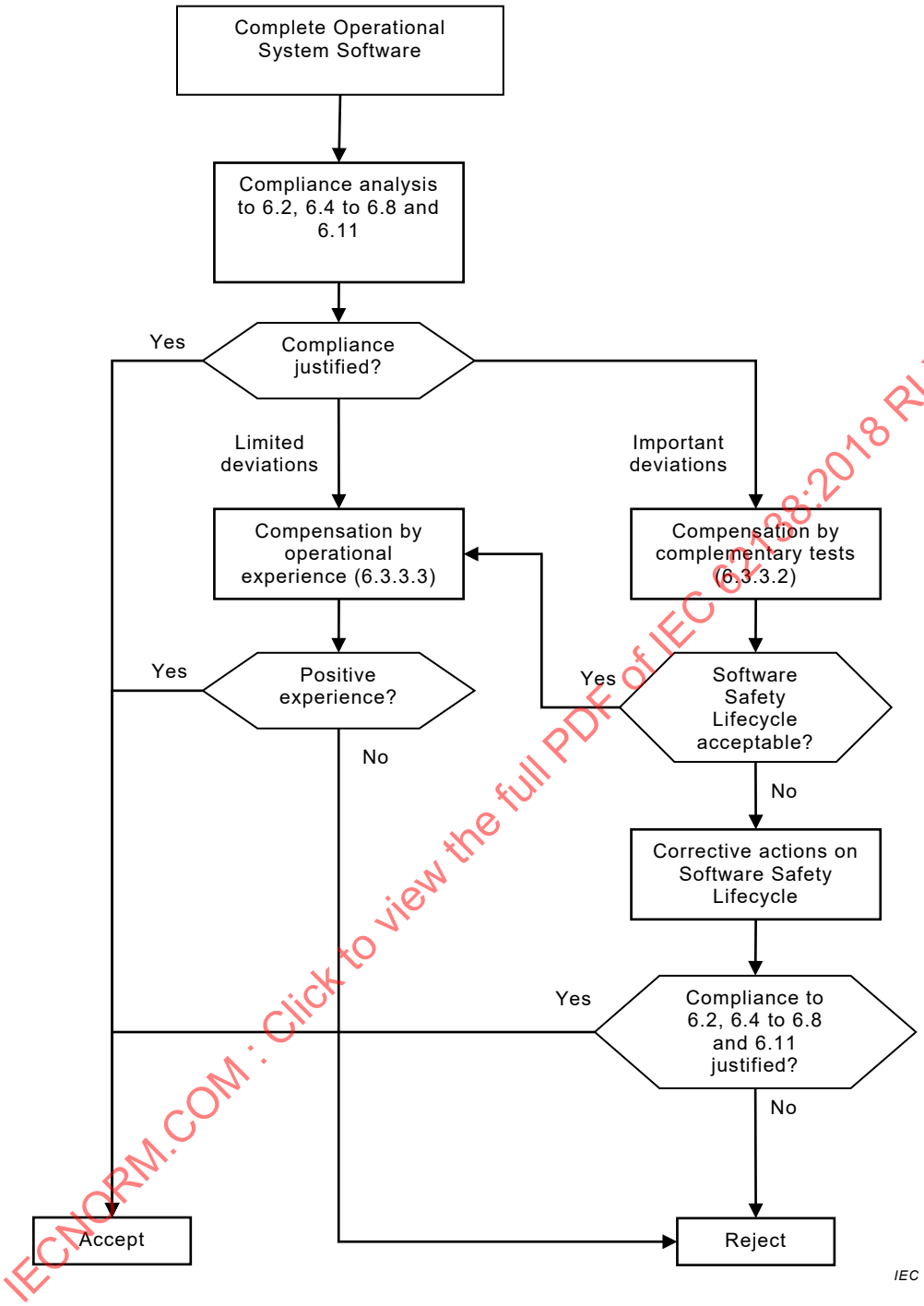


Figure 5 – Overview of the typical qualification process for pre-developed complete operational system software

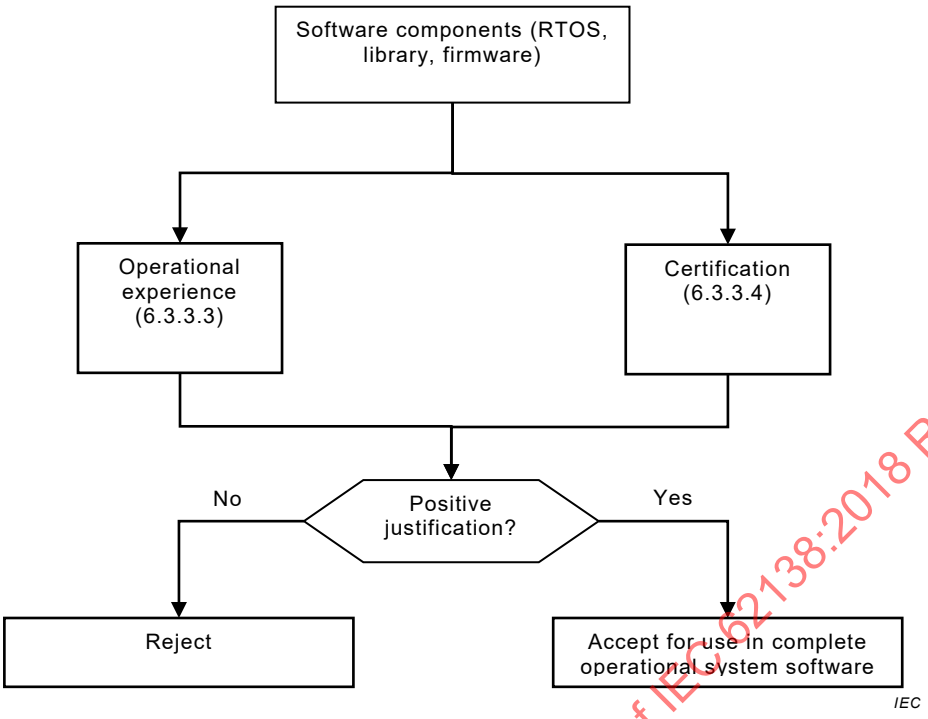


Figure 6 – Overview of the typical qualification process for pre-developed software components

6.3.3.1.8 To justify the correctness of complete operational system software with respect to its documentation for safety, a compliance analysis with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) should be performed first.

6.3.3.1.9 When the compliance analysis shows that the complete operational system software complies with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) then it should be accepted.

When the compliance analysis shows that the complete operational system software has limited deviations with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) then these limited deviations may be compensated by relevant, sufficient and positive operational experience (see 6.3.3.3). In cases where positive operational experience is not available, complementary tests may be used.

Limited deviations are the cases where a full software safety lifecycle has been followed and documented for the complete operational system software but in the execution of the different phases not all the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) have been satisfied.

Concerning for instance the software requirements specification phase (6.4), the case where the software requirements of an I&C system have been specified and documented, but do not include all the contents required in 6.4.4, is an example of a limited deviation.

6.3.3.1.10 When the compliance analysis shows that the complete operational system software has deviations with 6.2.4 then they should be compensated for by relevant, sufficient and positive operational experience (see 6.3.3.3).

6.3.3.1.11 When the compliance analysis shows that the complete operational system software has important deviations with 6.2.2, 6.7 or 6.8, then they should be compensated for by complementary tests (6.3.3.2).

6.3.3.1.12 When the compliance analysis shows that the complete operational system software has important deviations with 6.2.1, 6.2.3, 6.2.5, 6.4, 6.5, 6.6 or 6.11 then the Software safety lifecycle is not acceptable. In such cases, corrective actions should be implemented successfully to accept the software. The goal of the corrective actions should be to achieve compliance with the general subclauses of this document (6.2, 6.4 to 6.6 and 6.11). If corrective actions are not possible, the complete operational system software should be rejected.

Concerning 6.3.3.1.11 and 6.3.3.1.12, important deviations are cases where a full software safety lifecycle has not been followed and documented. The case where a lifecycle has been followed but is not documented is to be interpreted as important deviation.

The case where no validation has been documented is an example of an important deviation.

6.3.3.1.13 The strategy to justify the correctness of pre-developed software with respect to its documentation for safety should be defined and agreed by all parties involved in the early stages of the development of the I&C system.

This strategy cannot be fully defined before the completion of the compliance analysis of the complete operational system software with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) as it depends on the gaps that need to be filled.

6.3.3.2 Complementary tests

6.3.3.2.1 General

Complementary tests may be used to support the justification of correctness of pre-developed software, under the following conditions:

6.3.3.2.2 The complementary tests performed on pre-developed software during the development of an I&C system shall be documented.

6.3.3.2.3 The complementary tests shall provide evidence that, in the conditions of use within the I&C system, the pre-developed software is, and behaves as specified by, its documentation for safety.

The conditions of use may concern aspects such as the configuration of the pre-developed software (particularly the setting of parameters and configuration data), the use of functions and interfaces, the hardware environment, the processor and the demand loads.

6.3.3.2.4 *For class 2, the rules used to design complementary tests should be documented and justified.*

6.3.3.2.5 The documentation of complementary tests shall record:

- the version concerned and~~, *when relevant,*~~ the configuration of the pre-developed software;
- a description of the tests performed and, when relevant, of the environment used, so as to allow these tests to be repeated in identical conditions;
- the ~~hypotheses~~ assumptions made to develop the tests, and the evidence of their validity;
- the results obtained, and evidence of their correctness;
- the conclusions reached and the resolutions agreed.

6.3.3.3 Operational experience

6.3.3.3.1 General

Operational experience in systems of a lower safety class, or in non-safety classified systems may be taken into consideration. Operational experience may be used to ~~complement~~ support the justification of correctness of pre-developed software, under the following conditions:

6.3.3.3.2 The volume of the operational experience taken into consideration shall be documented.

6.3.3.3.3 *For class 2, the operational experience taken into consideration shall correspond to precisely identified versions of the pre-developed software and, when this software is specific to equipment, of the equipment in which it operates.*

6.3.3.3.4 *For class 2, when all or part of the operational experience corresponds to other versions of the pre-developed software and/or of the equipment, the differences with the versions to be used in the I&C system shall be assessed, and the relevance of this operational experience shall be justified.*

6.3.3.3.5 *For class 2, documented justification shall be given that the operational experience taken into consideration corresponds to conditions of use covering those ~~in the I&C system, or were even more severe~~ of the I&C system (the intended configuration of the software is one of the conditions of use). In cases where the operational experience conditions are the same, experience in systems of a lower safety class, or in non-safety classified systems can be taken into consideration.*

~~*The volume of the operational experience taken into consideration shall be documented.*~~

6.3.3.3.6 *For class 2, the methods used for collecting the operational experience taken into consideration shall be documented. In particular, documented justification shall be given that the failures (if any) caused by the pre-developed software during the operational experience taken into consideration were correctly detected and reported.*

6.3.3.3.7 *For class 2, evidence shall be provided that these failures were correctly analysed, and that the corresponding software faults corrected.*

~~*Operational experience in systems of a lower safety class, or in non-safety classified systems may be taken into consideration, provided that the requirements of this Subclause are satisfied.*~~

6.3.3.4 Certification

6.3.3.4.1 General

Pre-developed software used in systems important to safety already in operation (albeit not necessarily in I&C systems of nuclear power plants) may have been certified for compliance to some safety documents. The evidence provided by such a certification may ~~be taken into consideration~~ strongly support the justification of correctness of pre-developed software under the following conditions:

6.3.3.4.2 The safety document used for the certification of the pre-developed software shall address explicitly the software development process.

6.3.3.4.3 The certification taken into consideration shall be documented.

6.3.3.4.4 The precise identification of the pre-developed software certified shall be documented. If it was certified as a part of a larger product (for example, as a part of an

equipment or equipment family), the precise identification of this product shall also be documented.

6.3.3.4.5 *For class 2, the evidence supporting the certification shall be assessable, in particular:*

- *the conditions (for example, the conditions of use and the assumptions) of the certification;*
- *the methods and tools used for the certification;*
- *the results obtained (for example, the properties and/or measurements certified).*

6.3.3.4.6 *For class 2, the relevance of these conditions and results to the evidence of correctness ~~and to the I&C system~~ shall be justified.*

6.3.3.4.7 *For class 2, the effectiveness of the methods and tools used for the certification should be justified.*

6.3.3.4.8 *For class 2, the certifying authority shall be identified and shall be competent for the properties and/or measurements certified.*

6.3.3.4.9 *For class 2, the version of the pre-developed software certified shall be the same as the one used in the I&C system.*

6.3.3.5 Modification

6.3.3.5.1 General

When a well-identified and limited modification is made to pre-developed software for which an appropriate justification of correctness already exists~~, or when it is necessary to remove faults,~~ the following requirements can be used as a substitute for the requirements of 6.3.3.1 to 6.3.3.4 to update or complete the justification. A change in the configuration data of the pre-developed software does not constitute a modification, provided that the new configuration remains within the range covered by the justification.

6.3.3.5.2 The modification of the pre-developed software shall be documented.

6.3.3.5.3 *For class 2, the documentation of the modification shall state:*

- *the precise identification of the modified software;*
- *the context of the modification, if the software is a part of a larger product (for example, an equipment or equipment family);*
- *the objectives, the specification and the constraints of the modification;*
- *the changes made to the documentation for safety.*

~~*It should also state the changes made to the design of the pre-developed software.*~~

6.3.3.5.4 *For class 3, the documentation of the modification should state:*

- *the precise identification of the modified software;*
- *the context of the modification, if the software is a part of a larger product (for example, an equipment or equipment family);*
- *the objectives, the specification and the constraints of the modification;*
- *the changes made to the documentation for safety.*

Concerning 6.3.3.5.3 and 6.3.3.5.4, the context of a modification may for example indicate:

- *the precise identification of the modified larger product;*
- *the objectives, the specification and the constraints of the modification of the product;*
- *the modifications in the rest of the product that need to be made or that may have an impact on the pre-developed software;*
- *the verification and validation actions performed at the level of the product.*

6.3.3.5.5 *For class 2, the documentation should also state the changes made to the design of the pre-developed software.*

6.3.3.5.6 Documented evidence (for example, based on manual inspections, tool supported analyses and/or tests) regarding the modified software, and possibly the larger product, shall justify that:

- the objectives of the modification are satisfied;
- no faults have been introduced;
- the modified software conforms to its updated documentation for safety.

6.3.3.5.7 *For class 2, the sufficiency of this evidence shall be justified, possibly taking into account the modifications made and the conditions of use within the I&C system.*

6.3.3.5.8 The documentation for safety shall be updated as required to maintain its accuracy with respect to any modifications to the software that could affect how the end user installs, operates or maintains the system of which the software is part of.

6.3.4 Functional suitability

6.3.4.1 General

The objective of this subclause is to ensure that pre-developed software is well-suited for the needs of the I&C system, and that it is not too complex with respect to these needs.

6.3.4.2 When applicable, the documentation for safety of pre-developed software shall be evaluated with respect to the system specification and system design. Inconsistencies shall be resolved.

6.3.4.3 *For class 2, the functions of the pre-developed software which are not required to support the system requirements specifications should be identified. A justification ~~of harmlessness~~ that these functions do not have a detrimental effect on safety should be provided.*

~~**6.2.4 Selection and use of dedicated devices with embedded software**~~

~~Black-box devices with embedded software may be used in an I&C system under the following conditions.~~

~~1 The software of the device shall be integrated in such a way that it cannot be modified by the user and cannot be used separately from the rest of the device.~~

~~2 The complexity of the functions of the device, its potential for configuration, and the extent of its interfaces and interactions with the rest of the I&C system shall be limited so as to allow a thorough functional coverage by tests.~~

~~*3 Evidence shall be given that the device conforms to the requirements of 6.2.1, 6.2.2 and 6.2.3.*~~

~~4 Evidence shall be given that the configuration and the use of the device in the I&C system conforms to the requirements of 6.4, 6.5.1 and 6.5.2.~~

~~*5 The design of the I&C system and/or of its software shall ensure that the device is used in clearly defined conditions.*~~

6.3.5 Selection and use of digital devices of limited functionality

IEC 62671 may be used as an alternative to this document for digital devices of limited functionality. IEC 62671 contains precise criteria to determine if it is applicable to a particular device.

6.4 Software requirements specification

6.4.1 General

This Subclause 6.4 completes and adds precision to the requirements of 6.2.3.4 of IEC 61513:2011.

6.4.2 Objectives

6.4.2.1 The requirements for the software of an I&C system shall be specified and documented.

The corresponding document or set of documents is called the software requirements specification. In principle, its objective is to specify what the software is to achieve without specifying how it ~~must~~ shall do it. However, design and implementation constraints may have to be specified when this is required by considerations of the design of the I&C system or of the I&C architecture.

6.4.2.2 *For class 2, the software requirements specification should avoid unnecessary complexity of the software design~~, and should aim at providing stable conditions of use for the software~~.*

6.4.2.3 The software requirements specification shall be such that:

- it contributes to the confidence in the correctness of the design of the I&C system;
- compliance of the I&C system to the requirements of IEC 61513:2011 can be demonstrated.

The IEC 61513:2011 requirements concerned with software requirements specification are mainly in 6.2.2.3, 6.2.2.4, 6.2.2.5, 6.2.3.3, 6.2.3.5 and 6.2.4.

6.4.2.4 The software requirements specification shall be a reference for software design, software validation, and possible software modifications.

6.4.3 Inputs

6.4.3.1 *For class 2, the inputs to the software requirements specification shall include the system specification and the system design documentation.*

They may also include other documents, for example:

- *project specific constraints;*
- *applicable rules and standards;*
- *requirements such as independence between functions;*
- *integrity requirements such as self-supervision to drive outputs to a safe state in the event of detectable failures.*

6.4.3.2 *For class 2, the structure of the software requirements specification should facilitate verification to ensure that it is consistent and complete with respect to its input documents.*

The software requirements specification may reference ~~parts of its~~ input~~s~~ documentation directly, so as to avoid unnecessary duplications and minimise the risk of inconsistencies. It may also reference other pre-existing documents, such as the documentation of pre-developed software.

6.4.3.3 The software requirements specification shall provide traceability to its input documents.

6.4.3.4 The verification of the software requirements specification (see 6.2.2) should notably check that it is consistent and complete with respect to its input documents.

6.4.3.5 The references, if any, made by the software requirements specification to other documents shall be precise so as to be unambiguous.

6.4.3.6 *For class 2, the software requirements specification should avoid unnecessary ~~additions with respect to its inputs~~ functionality.*

In principle, it is preferable that the software does not have more capabilities than required so as to minimise complexity. However, because current industrial practice is based on the use of pre-developed components, the inclusion of non-required capability may be justified.

6.4.4 Contents

6.4.4.1 The software requirements specification shall specify:

- the application functions to be ~~provided~~ performed by the software;
- the different modes of ~~behaviour~~ operation of the software, and the corresponding conditions of transition;
- the interfaces and interactions of the software with its environment (for example, with operators, with the rest of the I&C system, with the other systems and equipment with which it interacts or shares resources), including the roles, types, formats, ranges and constraints of inputs and outputs;
- the parameters of the software which are to be modified by operators during operation, if any, their roles, types, formats, ranges and constraints, and the checks to be performed by the software when they are modified;
- required performance, when appropriate;
- what the software ~~must~~ shall not do or ~~must~~ shall avoid, when appropriate;
- the requirements of, or the assumptions to be made by, the software regarding its environment, when applicable.

6.4.4.2 The software requirements specification should also specify the conditions (for example, the demand load), in particular the worst case conditions, provided to the software by its environment.

Concerning 6.4.4.1 and 6.4.4.2, functions, interfaces and performances requirements may depend on the mode of ~~behaviour~~ operation, on the values of the parameters, on the configuration data and on the conditions provided to the software.

6.4.4.3 The software requirements specification shall specify the software modes of ~~behaviour~~ operation required when errors or failures are detected. When periodic tests are required of the I&C system, the software requirements specification shall also specify the mode of ~~behaviour~~ operation required when such tests are performed.

6.4.4.4 The software requirements specification ~~should state the software quality objectives and~~ shall state the constraints to be respected by software design and implementation for the sake of correctness and robustness.

For example, this may include constraints:

- to give confidence in the correctness of software and system design (for example, margins to be taken when using dynamically allocated resources such as memory, processing power, communication bandwidth, operating system resources);
- to enhance the ability of the software and of the I&C system to tolerate faults, to detect and signal errors and failures, to take specified modes of ~~behaviour~~ operation and to recover from failures;
- to give confidence that operator mistakes and failures of other systems or equipment with which the software interacts or shares resources will not lead to unacceptable effects.

6.4.4.5 The software requirements specification should state the expectations to be respected by software design and implementation for the sake of correctness and robustness.

6.4.4.6 The software requirements specification shall specify the contribution of the software to the assurance that the operators will be informed in due time of errors or failures concerning the functions of the I&C system identified as important to safety. The information provided to the operators shall allow them to take any appropriate action.

6.4.4.7 The software requirements specification shall identify the functions and the requirements related to safety category B or C.

6.4.5 Properties

6.4.5.1 *For class 2, the ~~languages~~ notations, rules and documents used to develop the software requirements specification should contribute to its clarity and precision, and should be chosen taking into account those used in the inputs and those chosen for the design and implementation of ~~new~~ software.*

Since any particular specification format does not always allow a clear, precise and verifiable expression of all specification needs, different and complementary formats may be used in the same software requirements specification. For example, application functions may be specified using a different format than those used for other functions.

6.4.5.2 *For class 2, the requirements of the software requirements specification shall be expressed in such a way that their satisfaction can be assessed objectively.*

6.5 Software design

6.5.1 Objectives

6.5.1.1 The design of software shall be documented. ~~The documentation should give an overview of the organisation and of the functioning of the software.~~

The corresponding document or set of documents is called the software design specification. When pre-developed software is used, the software design specification may make reference to the corresponding documentation.

6.5.1.2 The software design specification should give an overview of the organisation and of the functioning of the software (see also 6.5.3.3).

6.5.1.3 *For class 2, the software design specification shall contribute to the confidence in the quality of the design of the software, and in its correctness with respect to the software requirements specification.*

6.5.1.4 The software design specification shall provide evidence that the software requirements specification statements important to safety are taken into account ~~and will be satisfied~~ in all specified conditions.

6.5.1.5 *For class 2, the software design specification should document the measures taken by the software to ensure that any error or failure of the software is detected early and does not propagate beyond the limits it should specify. It should also document the actions that are taken when an error or failure is detected.*

6.5.1.6 The software design specification shall ensure, if applicable, that the adverse side effects of software errors and failures are cleared prior to returning to a normal mode of ~~behaviour~~ operation.

6.5.1.7 The software design shall be produced to achieve modularity, testability and maintainability.

6.5.1.8 *For class 2, providing this does not lead to excessive complexity, the design of the software of an I&C system should facilitate:*

- *the analysis and testing of the software and of its components;*
- *the localisation of faults;*
- *the identification of the effects of a modification.*

6.5.1.9 The software design specification shall be a reference for software implementation and integration, and for possible software modifications.

6.5.2 Inputs

6.5.2.1 The inputs to the software design process shall include the software requirements specification and the documentation for safety of pre-developed software.

They may also include other documents, such as project specific constraints, and/or applicable rules and standards.

6.5.3 Contents

6.5.3.1 The software design specification shall include the specification of:

- the overall organisation of the software;
- the overall functioning of the software under the conditions and modes of ~~behaviour~~ operation required by software requirements specification.

6.5.3.2 The overall organisation should provide information regarding:

- the precise identification and the configuration of pre-developed software;
- the distribution of resources, software components and software tasks over sub-systems;
- the allocation of software (sub-)functions ~~and performances~~ to the identified software tasks;
- the main internal interfaces, in particular the interfaces between software tasks.

6.5.3.3 The overall functioning should provide information regarding:

- interactions, communication protocols and information flows;
- sequencing and timing constraints;
- use of resources;
- synchronisation, particularly when using shared resources.

6.5.3.4 *For class 2, the software design specification shall document how the software requirements that are important to safety are met under all specified conditions. When pre-developed software is used, the demonstration regarding the software properties important to*

safety shall be based in particular on the predictive information provided by the corresponding documentation for safety (see 6.3.2.2.5).

6.5.3.5 *For class 2, the software design specification and the system design documentation shall state and justify the measures taken to mitigate the effects of the known or anticipated failure modes of any pre-developed software ~~or device with embedded software~~ for which complementary means for providing evidence of correctness have been used (see 6.3.3).*

6.5.3.6 *For class 2, the software design specification shall provide rules for software implementation.*

6.5.3.7 *For class 2, the software design specification should in particular specify rules for configuring and for using pre-developed software, so as to ensure that this software is used in a controlled way consistent with the corresponding documentation for safety.*

6.5.3.8 *For class 2, the software design specification shall include the detailed design of any ~~new~~ software implemented in general-purpose language.*

6.5.3.9 The software design specification of a component of ~~such~~ any software implemented in general-purpose language should specify:

- the functions to be provided by the component, with interfaces, roles, types, formats, ranges and constraints of inputs, outputs, exception signals, and configuration data;
- the required performance (for example, response time, accuracy), when appropriate;
- the requirements of the component regarding its environment (for example, needs in terms of dynamically allocated memory, operating system resources, etc.), when appropriate;
- any other information that the users of the component ~~must~~ shall be aware of;
- any relevant implementation constraint.

6.5.3.10 *For class 2, the software design specification shall provide information enabling correct predictions regarding the key safety significant elements of system performance, including notably the maximum response times and the maximum usage of resources.*

Such information may be provided in the form of data, formulae and/or models allowing the calculation of worst case response times and resources usage of applications.

6.5.4 Properties

6.5.4.1 *For class 2, the software design specification shall present the design of the software clearly and precisely. ~~The format and the syntax used to express the design should contribute to clarity and precision.~~*

6.5.4.2 *For class 3, the software design specification should present the design of the software clearly and precisely.*

Concerning 6.5.4.1 and 6.5.4.2, the main approach may be a top-down approach, but some documents may also give information that highlights how aspects of particular importance (for example, tolerance to failures) are taken into account across the software or across the I&C system.

6.5.4.3 *For class 2, the format and the syntax used to express the design in the software design specification should contribute to clarity and precision.*

6.6 Implementation of software

6.6.1 General requirements

6.6.1.1 General

The requirements of this subclause are applicable to all ~~new~~ software, i.e., to the configuration of pre-developed software, and to computer programs written in application-oriented or general–purpose languages.

6.6.1.2 The use of pre-developed software shall be verified to be consistent with the corresponding documentation for safety and with the constraints set by the software design specification.

6.6.1.3 The procedures used to translate ~~new~~ computer programs into executable code shall be documented and verified.

These procedures typically describe how the compiler tool chain or the code generator has to be invoked to translate computer programs into executable code. They are often automated.

6.6.1.4 *For class 2, the updating of executable code after changes in computer programs should be performed ~~whenever possible~~ by automated means.*

6.6.2 Configuration of software and of devices containing software

6.6.2.1 General

The requirement of this subclause is specific to the configuration of customisable software. Such software may be pre-developed or new. ~~The requirement is also applicable to the configuration of customisable devices with embedded software.~~ However, when the configuration data represents the sequencing of processing to be performed by the software or the system (i.e., it is effectively ~~software~~ computer programs), 6.6.3 applies.

6.6.2.2 The configuration of customisable software and devices with embedded customisable software shall be documented.

6.6.3 Implementation with application-oriented languages

6.6.3.1 General

The requirements of this subclause are specific to computer programs written in application-oriented languages. Generally, application oriented formats (such as logic diagrams or function block diagrams) may be used to express all or part of the software requirements specification or of the software design specification. Only limited detailed design and implementation effort is then necessary to transform the specification into computer programs that can be automatically translated into executable code or into a form suitable to be interpreted.

6.6.3.2 The parts of the software requirements specification and/or of the software design specification that are used to generate executable code by automated means shall be considered to be computer programs written in application-oriented languages.

6.6.3.3 *For class 2, computer programs written in application-oriented languages shall be verified to be functionally correct and consistent. The verification shall ensure that:*

- *all the design features are fully understood (i.e., there will be no unexpected behaviour under all specified conditions);*
- *the behaviour specified is consistent with the objectives set by the ~~inputs to Software Requirements Specification~~ software design specification.*

Animation, tests, reviews, walkthrough, formal analyses and proof may be applied to improve the understanding of specifications and to verify their functional correctness and consistency.

~~*Sufficiency criteria should be specified and documented for the testing of these programs, possibly taking into account the results of the other verification means. Justification should be given if these criteria are not met.*~~

~~*Such criteria may be based on functional and/or structural coverage measures.*~~

6.6.3.4 *For class 3, computer programs written in application-oriented languages which are related to functions important to safety shall be verified to be functionally correct and consistent.*

6.6.3.5 The tests shall be developed with respect to the functional requirements of the object under test and not solely to the internal structure of this object.

6.6.3.6 *For class 2, the functional coverage shall be justified prior to the execution of the tests, so that successful execution of the tests confirms the compliance of the object with all its required behaviours.*

6.6.3.7 *For class 2, during test execution, the structural coverage reached by the tests should be monitored with respect to justified criteria (e.g. statement, condition, branch, data flow) in order to ensure the absence of non-required behaviours. Justification should be given if these criteria are not met.*

6.6.3.8 *For class 2, computer programs written in application-oriented languages should conform to documented rules ~~designed to improve~~ aiming at clarity, modifiability and testability. Non-conformances should be justified.*

A set of rules may be specific to a language or to a set of computer programs. ~~Low complexity~~ Simplicity, clarity and standardisation of layout and presentation, modularity, presence of ~~pertinent~~ relevant comments, avoidance of the unsafe features of the language and of its tools are examples of properties that generally facilitate understanding, verification, testing and later modification.

6.6.4 Implementation with general-purpose languages

6.6.4.1 General

The requirements of this subclause are specific to computer programs written in general-purpose languages.

6.6.4.2 *For class 2, documented verification shall provide evidence that computer programs written in general-purpose languages conform to their specification as defined by the software design specification.*

This may consist of a combination of manual inspections, tool supported analyses, and/or tests.

Code reviews, walkthrough, check lists and other similar techniques are often powerful manual inspection methods that may be considered for identifying software faults.

Tool supported analyses may be used ~~to determine which programs are the most likely to contain faults, and/or~~ to prove formally that a computer program has (or does not have) given properties. For example, they may give assurance that, under given conditions (for example, that the inputs are within given ranges), the computer program or identified parts of the computer program do not contain certain types of faults (for example, use of non-initialised variables, arithmetic overflow or underflow).

Tests may be performed on the host hardware, or in a software engineering environment.

6.6.4.3 *For class 2, verification documentation shall record:*

- *the identity and the version of the computer programs concerned;*
- *all the information necessary to repeat the verifications in similar conditions;*
- *the ~~hypotheses~~ assumptions made, and the evidence of their validity;*
- *the results obtained, and evidence of their correctness;*
- *the conclusions reached and, in case of detected errors, the resolutions agreed;*
- *evidence of satisfaction of the ~~sufficiency~~ acceptance criteria.*

~~*Pass criteria shall be specified and documented for the testing of these programs, possibly taking into account the results of the other verification means. Justification shall be given when these criteria are not met.*~~

~~*Pass criteria may be based on functional and/or structural coverage measures.*~~

6.6.4.4 The tests shall be developed with respect to the functional requirements of the object under test and not solely to the internal structure of this object.

6.6.4.5 *For class 2, the functional coverage shall be justified prior to the execution of the tests, so that successful execution of the tests confirms the compliance of the object with all its required behaviours.*

6.6.4.6 *For class 2, during test execution, the structural coverage reached by the tests should be monitored with respect to justified criteria (e.g. statement, condition, branch, data flow) in order to ensure the absence of non-required behaviours. Justification should be given if these criteria are not met.*

6.6.4.7 Computer programs written in general-purpose languages shall conform to documented programming rules aiming at clarity, modifiability and testability. ~~These rules should be expressed so as to be verifiable, and should aim in particular at early detection and containment of software errors.~~

A set of rules may be specific to a language or to a set of computer programs. ~~Low complexity~~ Simplicity, structured programming, modularity, encapsulation, information hiding (so that users of a software item only have to concern themselves with the service that is provided rather than with the internal workings of the item), presence of ~~pertinent~~ relevant comments, avoidance of the unsafe features of the language and of its tools are examples of properties that may facilitate understanding, verification, test and modification.

6.6.4.8 *For class 2, the programming rules should be expressed so as to be verifiable, and should aim in particular at early detection and containment of software errors.*

6.6.4.9 *For class 2, when a static analysis tool can be used to analyse code complexity, then rules should specify acceptable metric limits.*

6.6.4.10 *For class 2, computer programs written in general-purpose languages shall be verified to be compliant with the applicable rules and standards. Non-conformances shall be justified, and appropriate counter-measures shall be taken, documented and justified where necessary.*

~~*Counter-measures in the case of non-conformance may be for example more thorough verification.*~~

For example a counter-measure may be the use of more thorough verification to check that the code is doing what it is intended to do.

6.7 Software aspects of system integration

6.7.1 General

The integration of software is considered as part of system integration. This subclause complements 6.2.5, 6.3.4 and 6.4.5 of IEC 61513:2011 by providing additional requirements specific, or of particular importance, to software.

6.7.2 Software integration and/or inspections shall show that the integrated system and the software:

- comply with the design provisions that ensure the satisfaction of the software requirements specification statements identified as important to safety;
- satisfy the constraints stated by the software requirements specification with respect to correctness and robustness.

6.7.3 *For class 2, when software validation testing ~~is not considered to have~~ has not sufficiently exercised the software, ~~then sufficient confidence in~~ evidence of correct operation of the software shall be obtained, either by performing additional software integration testing or by ~~other means~~ more thorough verification.*

6.7.4 Software integration shall be performed according to the provisions of the system integration plan or of a software integration plan.

6.7.5 Records of the application of the plan used for software integration shall be produced, for example, test results. In the event of software or system modifications being required, it shall be possible to repeat all, or a subset of, the integration tests to evaluate the extent of possible changes in behaviour.

6.7.6 *For class 2, traceability shall be provided between software design specification and the corresponding integration tests.*

6.7.7 *For class 3, traceability should be provided between software design specification and the corresponding integration tests.*

6.8 Software aspects of system validation

~~The objective of the validation of software is to ensure compliance of the integrated software with the functional, performance and interface specifications imposed by the I&C system requirements. Thus, it is considered as part of system validation. This Subclause complements 6.1.5 and 6.2.4 of IEC 61513 by providing additional requirements specific, or of particular importance, to software.~~

6.8.1 General

Aspects of software functionality are tested during system validation. This subclause complements 6.2.6, 6.3.5 and 6.4.6 of IEC 61513:2011 by providing additional requirements specific, or of particular importance, to software. Where discrepancies are revealed, validation may be continued with justification or may be stopped to correct the discrepancy before revalidation.

6.8.2 *For class 2, software validation shall show that, in the target I&C system, the integrated software conforms to each functional, performance and interface statement of the software requirements specification, and contributes as designed to the satisfaction of the system requirements specification. This shall include justification that:*

- *the specified software functions are correctly performed when their parameters and inputs are in the ranges specified by the software requirements specification, in the conditions of use defined in the software requirements specification;*
- *the system functions to which the software contributes are correctly performed in the conditions of use defined in the system requirements specification;*
- *the software provides defences as required by the software requirements specification against operator mistakes and failures of other systems and equipment;*
- *the software functions as expected in its different modes of operation;*
- *the plant engineering data used by, or integrated in, the I&C system is correct; in particular, the validation of the software shall show that this data ~~correctly describes and addresses~~ defines the interface between the systems and equipment of the plant with which the software interacts or shares resources;*
- *defences required ~~of~~ to be performed by the system ~~by~~ in the system requirements specification against operator mistakes and failures of other systems and equipment, and to which the software contributes, are correctly provided.*

~~NOTE For some aspects of validation testing, it may be acceptable to use a hardware platform identical to the actual final target platform if adequate justification is provided.~~

The validation tests are normally performed with the software integrated in the target I&C system. It may be acceptable to use a platform representative of the target I&C system to perform validation tests if adequate justification is provided.

The conditions of use of functions important to safety may include the concurrent operation of functions not important to safety notably the operation during high communication loading.

6.8.3 *For class 3, software validation shall show that, in the target I&C system, the integrated software conforms to the functional, performance and interface requirements that are identified as important to safety. This shall include justification that:*

- *the specified software functions important to safety are correctly performed when their parameters and inputs are in the ranges specified by the software requirements specification, in the conditions of use defined in the software requirements specification;*
- *the system functions important to safety to which the software contributes are correctly performed in the conditions of use defined in the system requirements specification;*
- *the software provides defences as required by the software requirements specification against operator mistakes and failures of other systems and equipment;*
- *the software functions as expected in its different modes of operation;*
- *the plant engineering data used by, or integrated in, the I&C system to implement functions important to safety is correct; in particular, the validation of the software shall show that this data defines the interface between the systems and equipment of the plant with which the software interacts or shares resources.*

The validation tests are normally performed with the software integrated in the target I&C system. It may be acceptable to use a platform representative of the target I&C system to perform validation tests if adequate justification is provided.

The conditions of use of functions important to safety may include operation during high communication loading.

6.8.4 Software validation ~~should~~ shall be performed according to the provisions of a plan that is preferably the system validation plan or a software validation plan. ~~If not, it shall be performed according to the provisions of a Software Validation Plan.~~

6.8.5 *For class 2, the plan used for software validation shall specify the validation actions to be performed, and shall show that all the functionality, performance and interface statements of the software requirements specification are correctly taken into account by*

these actions. It shall also specify the main phases of the software validation (for example, an off-site phase followed by an on-site phase) and the corresponding means, methods and tools to be used.

6.8.6 *For class 2, the plan used for software validation shall provide traceability between the software requirements specification and the corresponding validation actions.*

6.8.7 *For class 3, the plan used for software validation shall specify the validation actions to be performed, and shall show that all the functionality, performance and interface statements of the software requirements specification identified as important to safety are correctly taken into account by these actions. It shall also specify the main phases of the software validation (for example, an off-site phase followed by an on-site phase) and the corresponding means, methods and tools to be used.*

6.8.8 *For class 3, the plan used for software validation should provide traceability between the software requirements specification and the corresponding validation actions.*

6.8.9 Records of the application of the plan used for software validation shall be produced. In the event of software or system modifications being required, it shall be possible to repeat all, or a subset of, the validation tests to evaluate the extent of possible changes in behaviour.

6.8.10 *For class 2, the results of software validation shall be auditable by persons competent in the subjects addressed but not directly engaged in the validation process.*

6.8.11 *For class 3, the results of software validation should be auditable by persons competent in the subjects addressed but not directly engaged in the validation process.*

6.8.12 These records shall document the configuration of the software being validated and the configuration of the validation environment (for example, the hardware environment and the tools, if any).

6.8.13 The team that writes the plan used for software validation shall include at least one person who did not participate in the design and implementation.

6.9 Installation of software on site

6.9.1 General

Subclause 6.2.7 of IEC 61513:2011 provides requirements regarding the installation of the I&C system on site. This subclause provides additional requirements specific, or of particular importance, to the installation of software.

6.9.2 The procedure for installing software on site shall be documented. It shall guarantee that the correct and complete version of the software is installed.

6.9.3 The procedure for installing software on site shall include and specify on-site checks and tests to be performed before the I&C system is put into full operational use. In particular, the satisfaction of the conditions required for correct operation of the software shall be verified.

For example, these conditions may concern the hardware on which the software operates, or other systems with which the software interacts or shares resources.

6.10 Anomaly reports

6.10.1 If unexpected, apparently incorrect, unexplained or abnormal behaviour is ~~experienced~~ observed after acceptance into service, an anomaly report should be raised.

6.10.2 The anomaly report should give details of the behaviour, the software and hardware configurations and the activities in hand at the time. It should also include the originator, location, date, and a report identification.

~~The report identification may be added following an initial review of the report to ensure that it is valid.~~

6.10.3 The anomaly reports should be reviewed. Issues raised should be documented, tracked and resolved.

6.10.4 The anomaly should be reported to the designer and to the users.

6.11 Software modification

6.11.1 General

The decision to proceed with software modifications depends upon their impact on the I&C system. Therefore, they are subject to the requirements of 6.2.8 and 6.4.7 of IEC 61513:2011. This subclause provides additional requirements specific, or of particular importance, to software.

6.11.2 Software modifications shall be developed and verified so as to maintain consistency with the requirements of 6.2, 6.3, 6.4, 6.5 and 6.6. They shall be installed on-site in accordance with the requirements of 6.9.

6.11.3 Software modifications should be integrated and validated in a manner consistent with 6.7 and 6.8.

6.11.4 When the extent of a modification does not ~~necessitate a~~ require the full application of 6.7 and 6.8, the integration of the modified software shall be performed according to a regression software integration plan, and the validation shall be performed according to a regression software validation plan. The adequacy and thoroughness of these plans shall be justified taking into account the extent of any modifications made in the software requirements specification and in the software design specification. Records of the application of these plans shall be produced.

6.11.5 *For class 2, when the regression approach is used, the regression software integration plan and the regression software validation plan shall give adequate confidence that the modified software conforms in all respects to the ~~new~~ modified software requirements specification, and that:*

- *the objectives of the modification are satisfied;*
- *no fault is introduced;*
- *the modified and/or newly introduced pre-developed software behaves as specified by the corresponding documentation for safety and as expected by the modified software design specification;*
- *the other modified and/or new software components conform to their specification.*

6.11.6 Software modifications shall be comprehensively documented. In particular, all affected software documents shall be updated.

6.11.7 Software modification documentation should state:

- the objectives of the software modification, including any system-level objectives;
- ~~any changes made to its specification;~~
- ~~any constraints that need to be respected when developing the modification;~~

- the software components affected or created by the modification;
- identification of the versions of these components, both before and after modification.

The system level objectives of a modification are documented according to the requirements 6.4.7 of IEC 61513:2011.

6.11.8 *For class 2, software modification documentation should state in addition:*

- *any changes made to its specification;*
- *any constraints that need to be respected when developing the modification;*
- *the references of the modified design and/or implementation documents.*

~~*The system level context of a modification may for example indicate:*~~

- ~~*the version identification of the system (or equipment) both before and after modification;*~~
- ~~*the objectives, the specification and the constraints of the modification of the system;*~~
- ~~*the modifications in the rest of the I&C system and/or the other systems with which the software interacts, which need to be made or that may have an impact on the software;*~~
- ~~*the effects of the modification on the I&C system and/or the other systems with which the software interacts, on system operation, and on the data on site;*~~
- ~~*the constraints applicable to the installation of the modification on site.*~~

6.11.9 *For class 2, the level of detail of the documentation of a software modification shall be such that:*

- *it contributes as appropriate to the confidence in the correctness of the modified software and I&C system;*
- *compliance of the I&C system to the applicable requirements of IEC 61513:2011 can be demonstrated.*

The IEC 61513:2011 requirements that may be concerned are mainly in 6.2.2.3, 6.2.2.4, 6.2.2.5, 6.2.3.3, 6.2.3.5 and 6.2.4.

6.11.10 The effects of a software modification on the rest of the I&C system and on the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

6.11.11 The effects on software of modifications in the rest of the I&C system or in the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

6.12 Defences against common cause failure due to software

Systematic faults may be introduced in any design and implementation process due to human error. Therefore such faults may be introduced by errors or omissions in the system/software requirements specification or later during software design and implementation (either in the developed part or in an included pre-existing design). Systematic faults may also be introduced by software tools when such tools suffer themselves from systematic faults introduced in their design and implementation process. Software could therefore potentially be affected by latent systematic faults which could, under some triggering event, lead to the CCF of multiple instantiations of a software design.

The potential for CCF at system level is in the scope of higher level SC 45A Standards, in particular:

- IEC 61513:2011 5.4.2.6 that addresses defence against CCF;

- IEC 61513:2011 5.4.4.2 that addresses the assessment of reliability and defences against CCF.

This document defines development and verification processes and requirements which minimise the potential for software to have systematic faults and therefore, as such faults can cause CCF, also minimise the potential for CCF due to software.

Annex A
(informative)

Typical list of software documentation

Table A.1 gives a typical list of software documentation.

Table A.1 – Typical list of software documentation

References to the subclauses of this document	References
Documents relating to software production	
Software quality assurance plan*	6.2.1
Software verification plan	6.2.2
Software configuration management plan*	6.2.3
Documentation for safety of pre-developed software	6.3
Software requirements specification	6.4
Software design specification	6.5
Programming rules	6.6.3.8, 6.6.4.7
Software verification report	6.2.2, 6.6.3, 6.6.4
Software integration plan*	6.7
Software integration report*	6.7
Software validation plan*	6.8
Software validation report*	6.8
Software installation procedure on site*	6.9
Documents relating to anomaly	
Anomaly report	6.10
Documents relating to software modification	
Software modification documentation	6.11
Regression software integration plan**	6.11
Regression software integration report**	6.11
Regression software validation plan**	6.11
Regression software validation report**	6.11

* These documents may be omitted when their content is included in system documents, for example in the system quality assurance plan, the system configuration management plan, the system integration plan, the system integration report, the system validation plan, the system validation report or the system installation procedure on site.

** When the extent of a modification does not require the full application of 6.7 and 6.8. Subclauses 6.2, 6.3, 6.4, 6.5, 6.6 and 6.9 are always applicable to software modifications and consequently the documents relating to these subclauses have to be kept up to date for any modification.

Annex B
(informative)

Correspondence between IEC 61513:2011 and this document

Table B.1 shows correspondence between IEC 61513:2011 and this document.

Table B.1 – Correspondence between IEC 61513:2011 and this document

IEC 61513:2011 Subclauses		Subclause in this document
5.4.2.5	Tools	6.2.4
5.4.2.6	Defence against CCF	6.12
5.4.4.2	Assessment of reliability and defences against CCF	
5.6.2	Architectural design documentation	6.2.4
6	System safety life cycle, Figure 5	5.4
6.2.2.3.3	Internal behaviour of the system	6.3.2, 6.5
6.2.2.7	Qualification	6.2.4
6.2.3.2	Selection of pre-existing components	6.3
6.2.3.4	Software specification	6.4
6.2.4	System detailed design and implementation	6.5, 6.6
6.2.5	System integration	6.7
6.2.6	System validation	6.8
6.2.7	System installation	6.9
6.2.8	System design modification	6.11
6.3.2	System quality assurance plan	6.2.1
6.3.2.3	System configuration management plan	6.2.3
6.3.4	System integration plan	6.7
6.3.5	System validation plan	6.8
6.4.4	System detailed design documentation	6.5, 6.6
6.4.5	System integration documentation	6.7
6.4.6	System validation documentation	6.8
6.4.7	System modification documentation	6.11
6.5.3.3	Software evaluation and assessment	All
8.2	Requirements on the objectives to be achieved	All

Annex C
(informative)

Relations of this document with IEC 61508

C.1 General

This annex establishes the correspondence between this document and IEC 61508-3:2010.

At the system level, IEC 61513:2011, Annex D establishes the correspondence with IEC 61508-1:2010, IEC 61508-2:2010 and IEC 61508-4:2010.

C.2 Comparison of scope and concepts

IEC 61508 refers to “safety-related systems” in general while this document follows IAEA practice and refers to “systems important to safety” (i.e. important to nuclear safety).

IEC 61508 grades the safety integrity level required for a computer based system according to the risk reduction the system is required to provide. This is arrived at by determining the severity of the risk associated with the hazard, and assessing the frequency and consequences of the hazard and the protection to be provided by the system to reduce the risk from the hazard to a tolerable level.

The nuclear industry has traditionally used primarily a deterministic method to determine the safety significance of a system and its impact on the severity of risk associated with possible discharge of activity.

IEC 61508 requires an independent functional safety assessment by individuals and organisations of experience and independence that rise with the SIL (see Part 1).

In the nuclear sector, the plant operators (who are ultimately responsible for ensuring nuclear safety) are generally in charge of ensuring that adequate functional safety assessment has been performed, but this process is often subject to relevant national nuclear regulations.

C.3 Correspondence between this document and IEC 61508-3:2010

Table C.1 – Correspondence between this document and IEC 61508-3:2010

IEC 62138		IEC 61508-3:2010	
5.4	Software and system safety lifecycles	7.1	General
6.2.1	Software safety lifecycle – Software quality assurance		
6.2.2	Verification	7.9	Software verification
6.2.3	Configuration management	6.2.3	Software configuration management
6.2.4	Selection and use of software tools	7.4.4	Requirements for support tools, including programming languages
6.2.5	Selection of languages		
6.3	Selection of pre-developed software	7.4.2	General requirements
6.3.2	Documentation for safety		Annex D (normative) Safety manual for compliant items – additional requirements for software elements
6.4	Software requirement specification	7.2	Software safety requirements specification
6.5	Software design	7.4	Software design and development
6.6	Implementation of software		
6.7	Software aspects of system integration	7.5	Programmable electronics integration (hardware and software)
6.8	Software aspects of system validation	7.3	Validation plan for software aspects of system safety
		7.7	Software aspects of system safety validation
6.9	Installation of software on site		Outside the scope of IEC 61508-3 as it is addressed in IEC 61508-1
6.10	Anomaly reports		Outside the scope of IEC 61508-3 as it is addressed in IEC 61508-1
6.11	Software modification	7.6	Software operation and modification procedures
		7.8	Software modification
6.12	Defences against common cause failure due to software		IEC 61508-3 addresses defences against common cause failure due to software, in particular in annex C and annex F.
	In the nuclear sector, this assessment is connected to the licensing process and depends on the safety bodies and national regulations.	8	Functional safety assessment IEC 61508-1 imposes requirements for technical knowledge and independence of the functional safety assessor(s) graded on the basis of level of innovation, technical novelty and possible consequences of failures. In addition, most organizations that offer functional safety assessor(s) and product certifications are now accredited by national accrediting agencies.

NOTE Informative annexes of IEC 62138 and IEC 61508 are not considered in Table C.1.

Bibliography

~~IEC 60880, *Software for computers in the safety systems of nuclear power stations*~~

~~IEC 60880-2:2000, *Software for computers important to safety for nuclear power plants – Part 2: Software aspects of defence against common cause failures, use of software tools and of pre-developed software*~~

IEC 61508-3:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software requirements*

IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*

IEC 61511-1:2016, *Functional safety – Safety instrumented systems for the process industry sector – Part 1: Framework, definitions, system, hardware and ~~software~~ application programming requirements*

IEC 62645:2014, *Nuclear power plants – Instrumentation and control systems – Requirements for security programmes for computer-based systems*

ISO/IEC 12207:2008, ~~*Information technology*~~ *Systems and software engineering – Software life cycle processes*

ISO 9001:2015, *Quality management systems – Requirements*

ISO 90003:2014, ~~*Quality management and quality assurance standards – Part 3*~~ *Software engineering – Guidelines for the application of ISO 9001:~~*~~1994~~* *2008 to ~~the development, supply, installation and maintenance of~~ computer software*

~~IAEA Safety Standards Series, N° NS-R-1, *Safety of Nuclear Power Plants: Design – Requirements*~~

~~IAEA Safety Standards Series, N° NS-G-1.1, *Software for Computer-Based Systems Important to Safety in Nuclear Power Plants – Safety Guide*~~

~~IAEA Safety Standards Series, N° NS-G-1.3, *Instrumentation and Control Systems Important to Safety in Nuclear Power Plants – Safety Guide*~~

~~IAEA Safety Series, N° 50-C/SG-Q, *Quality Assurance for Safety in Nuclear Power Plants and Other Nuclear Installations – Code and Safety Guides Q1-Q14*~~

IAEA Safety Standard Series No. SSR-2/1:2016, *Safety of Nuclear Power Plant: Design*

IAEA Safety Guide SSG-39:2016, *Design of instrumentation and control systems in Nuclear Power Plants*

IAEA Safety Glossary:2016, *Terminology used in nuclear safety and radiation protection*

IAEA Safety Standard Series, N° GS-G-3.5:2009, *the Management System for Nuclear Installations*

IEEE Std 7-4.3.2:2010, *IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations*

DO-178 revision C:2012, *Software Considerations in Airborne Systems and Equipment Certification*

_____

IEC 62138

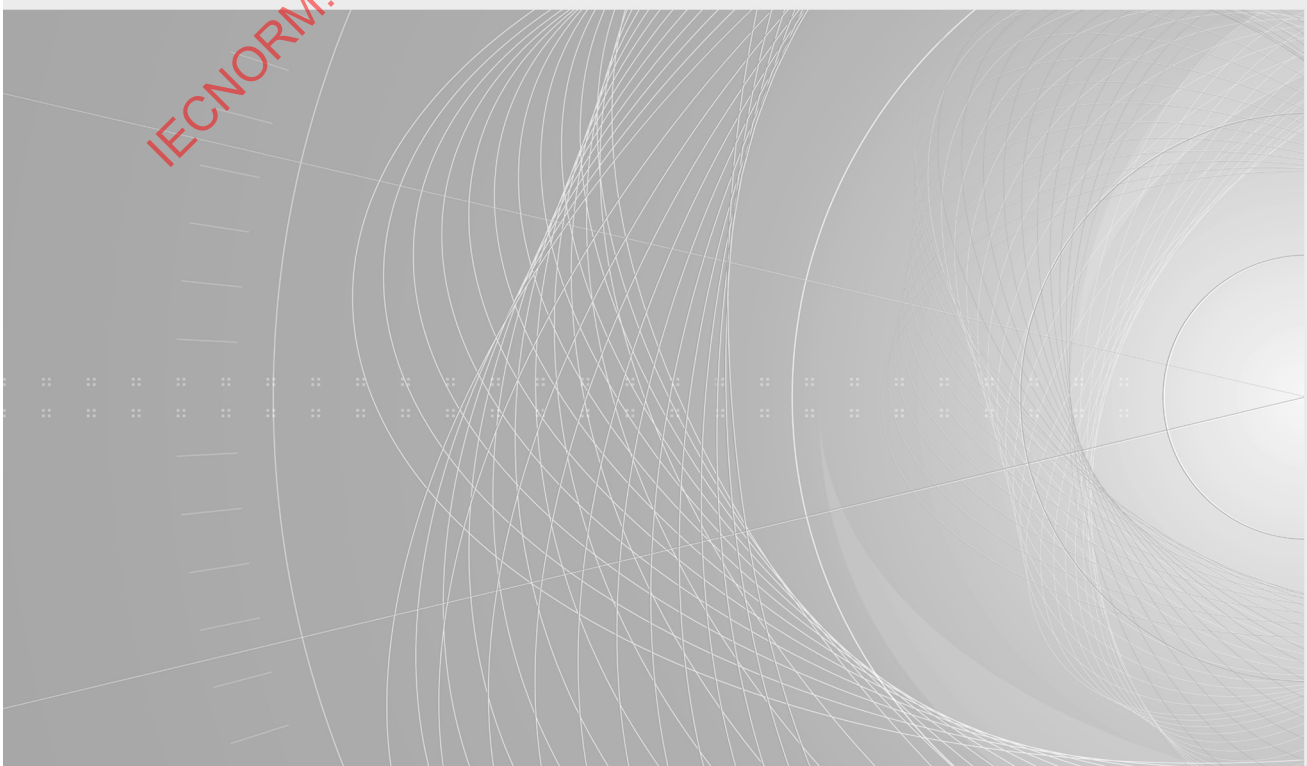
Edition 2.0 2018-07

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category B or C functions

Centrales nucléaires de puissance – Systèmes d'instrumentation et de contrôle-commande importants pour la sûreté – Aspects logiciels des systèmes informatisés réalisant des fonctions de catégorie B ou C



IEC 62138:2018-07(en-fr)

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

NUCLEAR POWER PLANTS – INSTRUMENTATION AND CONTROL SYSTEMS IMPORTANT TO SAFETY – SOFTWARE ASPECTS FOR COMPUTER-BASED SYSTEMS PERFORMING CATEGORY B OR C FUNCTIONS

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62138 has been prepared by subcommittee 45A: Instrumentation, control and electrical power systems of nuclear facilities, of IEC technical committee 45: Nuclear instrumentation.

This second edition cancels and replaces the first edition published in 2004. This edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

a) align the standard with standards published or revised since the first edition, in particular IEC 61513, IEC 60880, IEC 62645 and IEC 62671;
b) merge Clause 5 and Clause 6 of the first edition into a single clause in order to avoid the repetition of the vast majority of the text which proves to be extremely difficult to maintain in consistency;

c) revise clause on the selection of pre-developed software based on experiences from the application of the first edition of the standard on industrial projects. More precise criteria are proposed for the evidence of correctness of pre-developed software;

d) introduce requirements on traceability in consistency with IEC 61513;

e) introduce an Annex A that gives a typical list of software documentation;

f) introduce an Annex B that establishes relationship between IEC 61513 and this document;

g) introduce an Annex C that establishes relationship between IEC 61508 and this document.

The text of this standard is based on the following documents:

FDIS	Report on voting
45A/1201/FDIS	45A/1209/RVD

Full information on the voting for the approval of this International Standard can be found in the report on voting indicated in the above table.

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

In this document, the following print types are used:

- *Requirements and recommendations applicable specifically to class 2 or to class 3 systems appear in italics in Clause 6.*

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

INTRODUCTION

a) Technical background, main issues and organisation of this document

This International Standard provides requirements on the software aspects for computer-based instrumentation and control (I&C) systems performing category B or C functions as defined by IEC 61226. It complements IEC 60880 which provides requirements for the software of computer-based I&C systems performing category A functions.

It is consistent with, and complementary to, IEC 61513:2011. Activities that are mainly system level activities (for example, integration, validation and installation) are not addressed exhaustively by this document: requirements that are not specific to software are deferred to IEC 61513:2011.

This document takes into account the current practices for the development of software for I&C systems, in particular:

– the use of pre-developed software, equipment and equipment families that were not necessarily designed to nuclear industry sector standards;

– the use of application-oriented languages.

b) Situation of the current document in the structure of the IEC SC 45A standard series

IEC 61513 is a first level IEC SC 45A document and gives guidance applicable to I&C at system level.

IEC 62138 is a second level IEC SC 45A document that supplements IEC 61513 concerning software development of computer-based I&C systems performing category B or C functions.

For more details on the structure of the IEC SC 45A standard series, see item d) of this introduction.

c) Recommendations and limitations regarding the application of this document

This document is not intended to be used as a general-purpose software engineering guide. It applies to the software of I&C systems performing category B or C functions for new nuclear power plants as well as to I&C upgrading or back-fitting of existing plants.

For existing plants, only a subset of requirements is applicable and this subset has to be identified at the beginning of any project.

The purpose of the guidance provided by this document is to reduce, as far as possible, the potential for latent software faults to cause system failures, either due to single software failures or multiple software failures (i.e. Common Cause Failures due to software).

This document does not explicitly address how to protect software against those threats arising from malicious attacks, i.e. cybersecurity, for computer-based systems. IEC 62645 provides requirements for security programmes for computer-based systems.

To ensure that this document will continue to be relevant in future years, the emphasis has been placed on issues of principle, rather than specific technologies.

d) Description of the structure of the IEC SC 45A standard series and relationships with other IEC documents and other bodies documents (IAEA, ISO)

The top-level documents of the IEC SC 45A standard series are IEC 61513 and IEC 63046. IEC 61513 provides general requirements for I&C systems and equipment that are used to perform functions important to safety in nuclear power plants (NPPs). IEC 63046 provides general requirements for electrical power systems of NPPs; it covers power supply systems including the supply systems of the I&C systems. IEC 61513 and IEC 63046 are to be considered in conjunction and at the same level. IEC 61513 and IEC 63046 structure the IEC SC 45A standard series and shape a complete framework establishing general requirements for instrumentation, control and electrical systems for nuclear power plants.

IEC 61513 and IEC 63046 refer directly to other IEC SC 45A standards for general topics related to categorization of functions and classification of systems, qualification, separation, defence against common cause failure, control room design, electromagnetic compatibility, cybersecurity, software and hardware aspects for programmable digital

systems, coordination of safety and security requirements and management of ageing. The standards referenced directly at this second level should be considered together with IEC 61513 and IEC 63046 as a consistent document set.

At a third level, IEC SC 45A standards not directly referenced by IEC 61513 or by IEC 63046 are standards related to specific equipment, technical methods, or specific activities. Usually these documents, which make reference to second-level documents for general topics, can be used on their own.

A fourth level extending the IEC SC 45A standard series, corresponds to the Technical Reports which are not normative.

The IEC SC 45A standards series consistently implements and details the safety and security principles and basic aspects provided in the relevant IAEA safety standards and in the relevant documents of the IAEA nuclear security series (NSS). In particular this includes the IAEA requirements SSR-2/1, establishing safety requirements related to the design of nuclear power plants (NPPs), the IAEA safety guide SSG-30 dealing with the safety classification of structures, systems and components in NPPs, the IAEA safety guide SSG-39 dealing with the design of instrumentation and control systems for NPPs, the IAEA safety guide SSG-34 dealing with the design of electrical power systems for NPPs and the implementing guide NSS17 for computer security at nuclear facilities. The safety and security terminology and definitions used by SC 45A standards are consistent with those used by the IAEA.

IEC 61513 and IEC 63046 have adopted a presentation format similar to the basic safety publication IEC 61508 with an overall life-cycle framework and a system life-cycle framework. Regarding nuclear safety, IEC 61513 and IEC 63046 provide the interpretation of the general requirements of IEC 61508-1, IEC 61508-2 and IEC 61508-4, for the nuclear application sector. In this framework IEC 60880, IEC 62138 and IEC 62566 correspond to IEC 61508-3 for the nuclear application sector. IEC 61513 and IEC 63046 refer to ISO as well as to IAEA GS-R-3 and IAEA GS-G-3.1 and IAEA GS-G-3.5 for topics related to quality assurance. At level 2, regarding nuclear security, IEC 62645 is the entry document for the IEC SC 45A security standards. It builds upon the valid high level principles and main concepts of the generic security standards, in particular ISO/IEC 27001 and ISO/IEC 27002; it adapts them and completes them to fit the nuclear context and coordinates with the IEC 62443 series. At level 2, regarding control rooms, IEC 60964 is the entry document for the IEC SC 45A control rooms standards and IEC 62342 is the entry document for the IEC SC 45A ageing management standards.

NOTE 1 It is assumed that for the design of I&C systems in NPPs that implement conventional safety functions (e.g. to address worker safety, asset protection, chemical hazards, process energy hazards) international or national standards would be applied.

NOTE 2 IEC SC 45A domain was extended in 2013 to cover electrical systems. In 2014 and 2015 discussions were held in IEC SC 45A to decide how and where general requirement for the design of electrical systems were to be considered. IEC SC 45A experts recommended that an independent standard be developed at the same level as IEC 61513 to establish general requirements for electrical systems. Project IEC 63046 is now launched to cover this objective. When IEC 63046 is published, this NOTE 2 of the introduction will be suppressed.

NUCLEAR POWER PLANTS – INSTRUMENTATION AND CONTROL SYSTEMS IMPORTANT TO SAFETY – SOFTWARE ASPECTS FOR COMPUTER-BASED SYSTEMS PERFORMING CATEGORY B OR C FUNCTIONS

1 Scope

This document specifies requirements for the software of computer-based instrumentation and control (I&C) systems performing functions of safety category B or C as defined by IEC 61226. It complements IEC 60880 which provides requirements for the software of computer-based I&C systems performing functions of safety category A.

It is consistent with, and complementary to, IEC 61513. Activities that are mainly system level activities (for example, integration, validation and installation) are not addressed exhaustively by this document: requirements that are not specific to software are deferred to IEC 61513.

The link between functions categories and system classes is given in IEC 61513. Since a given safety-classified I&C system may perform functions of different safety categories and even non safety-classified functions, the requirements of this document are attached to the safety class of the I&C system (class 2 or class 3).

This document is not intended to be used as a general-purpose software engineering guide. It applies to the software of I&C systems of safety classes 2 or 3 for new nuclear power plants as well as to I&C upgrading or back-fitting of existing plants.

For existing plants, only a subset of requirements is applicable and this subset has to be identified at the beginning of any project.

The purpose of the guidance provided by this document is to reduce, as far as possible, the potential for latent software faults to cause system failures, either due to single software failures or multiple software failures (i.e. Common Cause Failures due to software).

This document does not explicitly address how to protect software against those threats arising from malicious attacks, i.e. cybersecurity, for computer-based systems. IEC 62645 provides requirements for security programmes for computer-based systems.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 60880:2006, *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions*

IEC 61226, *Nuclear power plants – Instrumentation and control important to safety – Classification of instrumentation and control functions*

IEC 61513:2011, *Nuclear power plants – Instrumentation and control important to safety – General requirements for systems*

IEC 62671:2013, *Nuclear power plants – Instrumentation and control important to safety – Selection and use of industrial digital devices of limited functionality*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1
animation
process by which the behaviour defined by a specification is displayed with actual values derived from the stated behaviour expressions and from some input values

[SOURCE: IEC 60880:2006, 3.1]

3.2
application function
function of an I&C system that performs a task related to the process being controlled rather than to the functioning of the system itself

[SOURCE: IEC 61513:2011, 3.1]

3.3
application software
part of the software of an I&C system that implements the application functions

Note 1 to entry: Application software contrasts with system software.

Note 2 to entry: Application software is plant specific, so it is not to be considered pre-developed software.

[SOURCE: IEC 61513:2011, 3.2 modified (modified notes to entry)]

3.4
application-oriented language
computer language specifically designed to address a certain type of application and to be used by persons who are specialists of this type of application

Note 1 to entry: Equipment families usually feature application-oriented languages so as to provide easy to use capability for adjusting the equipment to specific requirements.

Note 2 to entry: Application-oriented languages may be used to specify the functional requirements of an I&C system, and/or to specify or design application software. They may be based on texts, on graphics, or on both.

Note 3 to entry: Examples: function block diagram languages, languages defined by IEC 61131-3.

Note 4 to entry: See also general-purpose language.

[SOURCE: IEC 60880:2006, 3.3 modified (addition of note 4 to entry)]

3.5
common cause failure
CCF
failure of two or more structures, systems or components due to a single specific event or cause

Note 1 to entry: Common causes may be internal or external to an I&C system.

[SOURCE: IAEA Safety Glossary, 2016 edition]

3.6
complexity
degree to which a system or component has a design, implementation or behaviour that is difficult to understand and verify

[SOURCE: IEC 61513:2011, 3.9]

3.7
computer program
set of ordered instructions and data that specify operations in a form suitable for execution by a computer

Note 1 to entry: This includes traditional programs written in general-purpose languages. This also includes programs written in application-oriented languages.

[SOURCE: IEC 60880:2006, 3.10, modified (addition of note 1 to entry)]

3.8
computer-based item
item that relies on software instructions running on microprocessors or microcontrollers

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: A computer-based item is a kind of programmable digital item.

Note 3 to entry: This term is equivalent to software-based item.

3.9
configuration management
process of identifying and documenting the characteristics of a facility's structures, systems and components (including computer systems and software), and of ensuring that changes to these characteristics are properly developed, assessed, approved, issued, implemented, verified, recorded and incorporated into the facility documentation

[SOURCE: IAEA Safety Glossary, 2016 edition]

3.10
cybersecurity
set of activities and measures whose objective is to prevent, detect, and react to digital attacks that have the intent to cause:

- disclosures that could be used to perform malicious acts which could lead to an accident, an unsafe situation or plant performance degradation (confidentiality),
- malicious modifications of functions that may compromise the delivery or integrity of the required service by I&C CB&HPD systems (including loss of control) which could lead to an accident, an unsafe situation or plant performance degradation (integrity),
- malicious withholding or prevention of access to or communication of information, data or resources (including loss of view) that could compromise the delivery of the required service by I&C systems which could lead to an accident, an unsafe situation or plant performance degradation (availability).

Note 1 to entry: This definition is tailored with respect to the IEC 62645 scope, focusing on the prevention of, detection of and reaction to malicious acts by digital means on I&C CB&HPD systems. It is recognized that the term "cybersecurity" has a broader meaning in other standards and guidance, often including non-malevolent threats, human errors and protection against natural disasters, which are all out of the scope of IEC 62645.

[SOURCE: IEC 62645:2014, 3.6 modified (removal of note 2 to entry)]

3.11
dedicated functionality
property of devices that have been designed to accomplish only one clearly defined function or only a very narrow range of functions, such as, for example, capture and signal the value of a process parameter, or invert an alternating current power source to direct current. This function (or narrow range of functions) is inherent in the device, and not the product of programmability by the user

Note 1 to entry: Ancillary functions (e.g., self-supervision, self-calibration, data communication) may also be implemented within the device, but they do not change the fundamental narrow scope of applicability of the device.

Note 2 to entry: "Dedicated" in the sense in which it is used in IEC 62671 refers to design for one specific function that cannot be changed in the field.

[SOURCE: IEC 62671:2013, 3.7]

3.12
design specification
document or set of documents that describe the organisation and functioning of an item, and that are used as a basis for the implementation and the integration of the item

3.13
documentation for safety
document or set of documents that specifies how a product can be safely used for applications important to safety

Note 1 to entry: This definition is used in the context of pre-developed software (see 6.3).

3.14
dynamic analysis
process of evaluating a system or component based on its behaviour during execution. In contrast to static analysis

[SOURCE: IEC 60880:2006, 3.15]

3.15
electrical/electronic/programmable electronic item
E/E/PE item
item based on electrical (E) and/or electronic (E) and/or programmable electronic (PE) technology

Note 1 to entry: In this term and its definitions, the word "item" can be replaced by the words: system or equipment or device.

[SOURCE: IEC 61508-4:2010, 3.2.13, modified ("item" added and note to entry modified)]

3.16
equipment family
set of hardware and software components that may work co-operatively in one or more defined architectures (configurations). The development of plant specific configurations and of the related application software may be supported by software tools. An equipment family usually provides a number of standard functionalities (e.g. application functions library) that may be combined to generate specific application software

Note 1 to entry: An equipment family may be a product of a defined manufacturer or a set of products interconnected and adapted by a supplier.

Note 2 to entry: The term "equipment platform" is sometime used as a synonym of "equipment family".

[SOURCE: IEC 61513:2011, 3.17 modified (removal of note 1 to entry)]

3.17
error
discrepancy between a computed, observed or measured value or condition, and the true, specified or theoretical value or condition

Note 1 to entry: See also human error, fault, failure.

[SOURCE: IEC 61513:2011, 3.18, modified (addition of note 1 to entry)]

3.18
executable code
software that is included in the target system

Note 1 to entry: Executable code usually includes instructions to be executed by the hardware of the target system, and associated data.

3.19
failure
loss of the ability of a structure, system or component to function within acceptance criteria

Note 1 to entry: Equipment is considered to fail when it becomes incapable of functioning, whether or not it is needed at that time. A failure in, for example, a backup system may not be manifest until the system is called upon to function, either during testing or on failure of the system it is backing up.

Note 2 to entry: A failure is the result of a hardware fault, software fault, system fault, or operator or maintenance error, and the associated signal trajectory which results in the failure.

Note 3 to entry: See also human error, fault, error.

[SOURCE: IAEA Safety Glossary, edition 2016]

3.20
fault
defect in a hardware, software or system component

Note 1 to entry: Faults may be originated from random failures, that result e.g. from hardware degradation due to ageing, and may be systematic faults, e.g. software faults, which result from design errors.

Note 2 to entry: A fault (notably a design fault) may remain undetected in a system until specific conditions are such that the result produced does not conform to the intended function, i.e. a failure occurs.

Note 3 to entry: See also human error, error, failure.

[SOURCE: IEC 61513:2011, 3.21, modified (note 3 to entry modified)]

3.21
firmware
software which is closely coupled to the hardware characteristics on which it is installed. The presence of firmware is generally “transparent” to the user of the hardware component and, as such, may be considered to be effectively an integral part of the hardware design (a good example of such software being processor microcode). Generally, firmware may only be modified by a user by replacing the hardware components (for example, processor chip, card, EPROM) which contain this software with components which contain modified software (firmware). Where this is the case, configuration control of the hardware components of the equipment effectively provides configuration control of the firmware. Firmware, as considered by IEC 60987, is effectively software that is built into the hardware

[SOURCE: IEC 60987:2007, 3.4]

3.22
functional validation
verification of the correctness of the application functions specifications against the top level plant functional and performance requirements. It is complementary to the system validation that verifies the compliance of the system with the functions specification

[SOURCE: IEC 61513:2011, 3.23]

3.23
general-purpose language
computer language designed to address all types of usage

Note 1 to entry: The system software of equipment families is usually implemented using general-purpose languages.

Note 2 to entry: Examples: Ada, C, Pascal.

Note 3 to entry: See also application-oriented language.

[SOURCE: IEC 60880:2006, 3.20 modified (note 3 to entry added)]

3.24
human error (or mistake)
human action that produces an unintended result

Note 1 to entry: See also fault, error, failure.

[SOURCE: IEC 61513:2011, 3.26 modified (note 1 to entry added)]

3.25
I&C architecture
organisational structure of the I&C systems of a plant which are important to safety

[SOURCE: IEC 61513:2011, 3.27]

3.26
I&C system
system, based on E/E/PE items, performing plant I&C functions as well as service and monitoring functions related to the operation of the system itself

Note 1 to entry: The term is used as a general term which encompasses all elements of the system such as internal power supplies, sensors and other input devices, data highways and other communication paths, interfaces to actuators and other output devices. The different functions within a system may use dedicated or shared resources.

Note 2 to entry: The elements included in a specific I&C system are defined in the specification of the boundaries of the system.

Note 3 to entry: See also the definition of E/E/PE item and the associated notes.

Note 4 to entry: According to their typical functionality, IAEA distinguishes between automation / control systems, HMI systems, interlock systems and protection systems.

3.27
integration
progressive aggregation and verification of components into a complete system

3.28
library
collection of related software elements that are grouped together, but which are individually selected for inclusion in the final software product

[SOURCE: IEC 60880:2006, 3.24]

3.29
mode of operation
functional state of an item where it provides a specific operational behaviour

EXAMPLE: Initialisation mode, normal mode, degraded modes to be taken in case of error in the item.

3.30
operational system software
software running on the target processor during system operation

EXAMPLE: Operating system, input/output drivers, exception handler, communication software, application-software libraries, self-supervision, redundancy and graceful degradation management.

3.31
parameter
data item governing the behaviour of the I&C system and/or of its software, and that may be modified by operators during plant operation

3.32
pre-developed software
software that already exists, is available as a commercial or proprietary product, and is being considered for use

Note 1 to entry: In this document, pre-developed softwares are divided in two different types:

a) complete operational system software,

b) software components.

Note 2 to entry: Pre-developed software may be divided into software that has not been specifically developed for a specific hardware environment, and software integrated in hardware components that has to be used in association with this hardware.

Note 3 to entry: In this document, this term does not cover software tools, even when they are pre-developed.

Note 4 to entry: Application software is plant specific, so it is not to be considered pre-developed software.

[SOURCE: IEC 60880:2006, 3.28 modified (notes to entry added)]

3.33
pre-existing items
hard- or software or software-based equipment that already exists, is available as a commercial or proprietary product, and is being considered for use

Note 1 to entry: This definition is included for the consistency of the terms and definitions with IEC 61513:2011, but not used in this document, dedicated to software, the term pre-developed software is used.

[SOURCE: IEC 61513:2011, 3.36 modified (note 1 to entry modified)]

3.34
programmable digital item
item that relies on software instructions or programmable logic to accomplish a function

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: The main kinds of programmable digital items are computer-based items and programmable logic items.

Note 3 to entry: This term used by IEC SC 45A is equivalent to programmable electronic item (PE item) defined according to IEC 61508.

3.35
programmable logic item
item that relies on logic components with an integrated circuit that consists of logic elements with an inter-connection pattern, parts of which are user programmable

Note 1 to entry: In this term and its definition, the term item can be replaced by the terms: system or equipment or device.

Note 2 to entry: A programmable logic item is a kind of programmable digital item.

Note 3 to entry: See also the definition of E/E/PE item and the associated notes.

3.36
self-supervision
automatic testing of system hardware performance and software consistency of a computer-based I&C system

[SOURCE: IEC 60671:2007, 3.8]

3.37
software
programs (i.e. sets of ordered instructions), data, rules and any associated documentation pertaining to the operation of a computer-based I&C system

[SOURCE: IEC 61513:2011, 3.51]

3.38
software component
one of the parts that make up a complete software. Software components need to be integrated to form complete software

Note 1 to entry: In this document, a pre-developed software item can be considered a software component only if it is integrated in larger software to form complete operational system software. In particular, verification and validation of the complete operational system software has to be performed with the software components embedded. The integration may be within software that runs on a single processor, for example for Real Time Operating Systems or libraries. The integration may also be within software that run in close cooperation on several processors, for example the firmware of communication modules or input/output modules.

3.39
software development
all activities of the software lifecycle that lead to the creation of the software of an I&C system or of a software product and that cover all the phases from software requirements specification to validation and installation on site

3.40
software modification
change in an already agreed document (or documents) leading to an alteration of the executable code

Note 1 to entry: Software modifications may occur either during initial software development (for example, to remove faults found in later stages of development), or after the software is already in service.

[SOURCE: IEC 60880:2006, 3.36]

3.41
software safety lifecycle
necessary activities involved in the development and operation of the software of an I&C system important to safety occurring during a period of time that starts with the software requirements specification and finishes when the software is withdrawn from use

[SOURCE: IEC 60880:2006, 3.37]

3.42
software validation
test and evaluation of integrated software to ensure compliance with the functional, performance and interface specifications imposed by the I&C system requirements

Note 1 to entry: In this document, software validation is considered a part of system validation.

3.43
static analysis
process of evaluating a system or component based on its form, structure, content or documentation. In contrast to dynamic analysis

[SOURCE: IEC 60880:2006, 3.40]

3.44
system software
software designed for a specific computer system or family of computer systems to facilitate the operation and maintenance of the computer system and associated programs, for example, operating systems, computers, utilities. System software is usually composed of operational system software and support software

Note 1 to entry: Operational system software: software running on the target processor during system operation, such as: operating system, input/output drivers, exception handler, communication software, application-software libraries, self-supervision, redundancy and graceful degradation management.

Note 2 to entry: Support software: software that aids in the development, test, or maintenance of other software and of the system such as compilers, code generators, graphic editor, off-line diagnostic, verification and validation tools, etc.

Note 3 to entry: See also application software.

[SOURCE: IEC 61513:2011, 3.58 modified (notes 2, 3 and 4 to entry added)]

3.45
system validation
confirmation by examination and provision of other evidence that a system fulfils in its entirety the requirement specification as intended (functionality, response time, fault tolerance, robustness)

Note 1 to entry: The 2016 edition of the IAEA Safety Glossary gives the two following definitions:

Validation: The process of determining whether a product or service is adequate to perform its intended function satisfactorily. Validation may involve a greater element of judgment than verification.

Computer system validation: The process of testing and evaluating the integrated computer system (hardware and software) to ensure compliance with the functional, performance and interface requirements.

Firstly, the definition “system validation” is a specific case of validation. It refers to a specific product, namely to the validation of an I&C system. This is consistent with the IAEA definition. Secondly, the IEC definition specifies the reference of validation, namely the requirement specification whereas the IAEA definition only refers to the “intended function”.

[SOURCE: IEC 61513:2011, 3.59]

3.46
systematic fault
fault related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors

[SOURCE: IEC 61513:2011, 3.60]

3.47
verification
confirmation by examination and by provision of objective evidence that the results of an activity meet the objectives and requirements defined for this activity

[SOURCE: IEC 61513:2011, 3.62]

4 Symbols and abbreviated terms

CB	Computer-based
CCF	Common cause failure
EPROM	Erasable Programmable Read Only Memory
HMI	Human machine interface
HDL	Hardware Description Language
HPD	HDL-Programmed Device
I&C	Instrumentation and control
NPP	Nuclear power plant

5 Key concepts and assumptions

5.1 General

Clause 5 presents some of the key concepts and assumptions about the nature and the development of the software of I&C systems of safety class 2 or 3, upon which the normative text is based.

5.2 Types of software

Figure 1 illustrates the range of services offered by software in a typical I&C system or I&C architecture. Software may often be defined as being either system software or application software. System software may also be divided into operational system software, which is embedded in safety classified I&C systems, and support system software (or software tools) which is either off-line or embedded in non-safety classified support systems. Software may also be found in dedicated devices such as sensors and actuators, communication devices and Uninterruptible Power Supplies (UPSs).

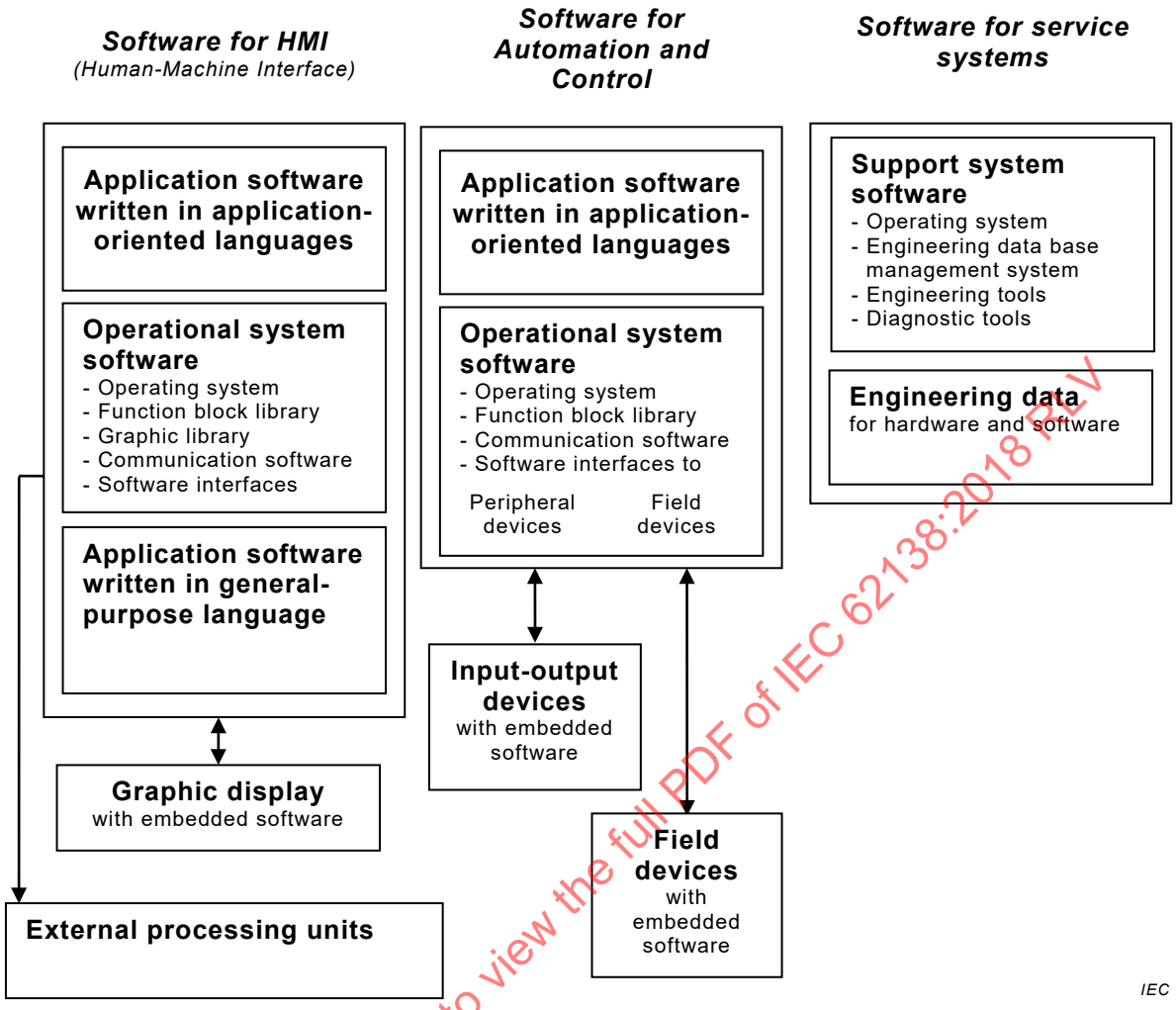


Figure 1 – Typical software parts in a computer-based I&C system

The software in an I&C system may also be divided into pre-developed software (which usually provides functions useful to a range of I&C systems) and new software (which is developed to the specific needs of the I&C system). The requirements of this document which address issues that are relevant to new software may also be applied retrospectively to pre-developed software. In some instances, however, this document provides alternative requirements specifically to address issues relevant to pre-developed software.

Many modern equipment families are provided with extensive application-oriented development tools that enable plant or system engineers to specify their requirements using graphical techniques. The tools may automatically translate the graphics representing computer programs into executable application software. When these tools are of adequate quality, this approach is considered to reduce the risk of faults.

5.3 Types of configuration data

Many system designs make extensive use of configuration data. Configuration data may be associated with operational system software or with application software. Configuration data associated with application software consists mainly of plant engineering data resulting from the design of the plant, and is often prepared by plant designers who are not required to have software skills. Configuration data may be divided into:

- data items which are not intended to be modified on-line by plant operators, and which are submitted to the same requirements as apply to the rest of the software;

- parameters, i.e., data items which may be modified by operators during plant operation (for example, alarm limits, set points, data required to calibrate instrumentation) and which need specific requirements.

5.4 Software and system safety lifecycles

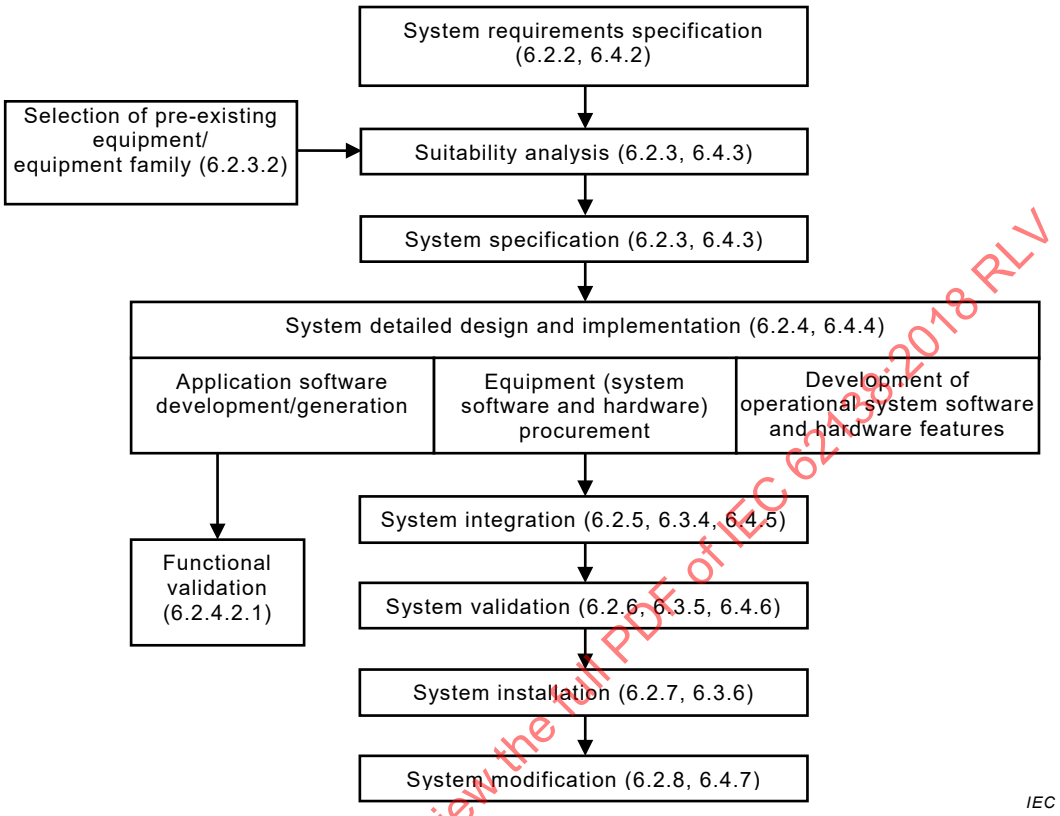
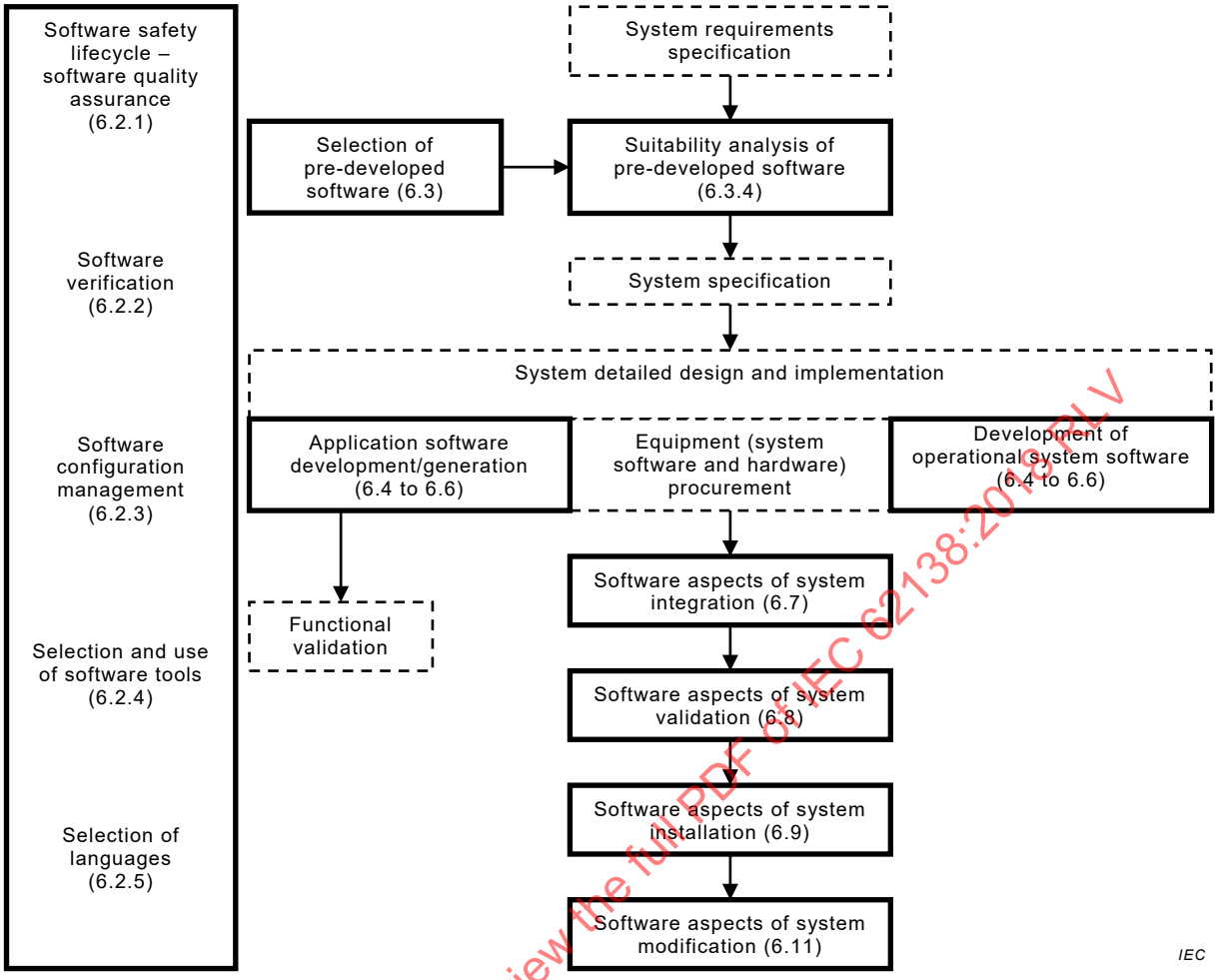


Figure 2 – Activities of the system safety lifecycle (as defined by IEC 61513:2011)

Software usually contributes strongly to the functions performed by the I&C system. It may also support additional functions needed for the operation of the system itself (for example, initialisation and supervision of hardware, communication between, and synchronisation of, sub-systems). Thus, the software safety lifecycle is in most cases strongly integrated with the system safety lifecycle. In particular, the software requirements specification is a part of, or is derived directly from, system specification and system design.

Although the verification of software is definitely a part of the software safety lifecycle, there is often no separate and well-identified boundary between software integration and system integration. Therefore, in this document, software integration is considered to be a part of system integration. Software validation too is considered a part of system validation.



NOTE Boxes in thin dotted lines represent system activities not addressed in this document.

Figure 3 – Software related activities in the system safety lifecycle

Figure 2 and Figure 3 illustrate the relationship between the activities of the software safety lifecycle and the activities of the system safety lifecycle.

It has to be noted that although IEC 61513:2011 identifies two different paths for the implementation of software (application software and operational system software, see Figure 2 and Figure 3), this document organises the requirements regarding the implementation of software into four subclauses:

- 6.6.1 provides requirements that are applicable whatever implementation technique is used;
- 6.6.2 provides requirements specific to the configuration of pre-developed software and of devices containing software, and in particular the setting of parameters and other configuration data;
- 6.6.3 provides requirements specific to the implementation and verification of software in application-oriented languages;
- 6.6.4 provides requirements specific to the implementation and verification of software in general-purpose languages.

As boxes titled “Application software development/generation” and “Development of operational system software” represent a large and essential part of the software safety lifecycle, a zoom is provided in Figure 4 which illustrates in more detail the activities between software requirements specification and software validation, with a clear representation of the

three different implementation paths (configuration of pre-developed software and devices, use of application-oriented languages and use of general-purpose languages).

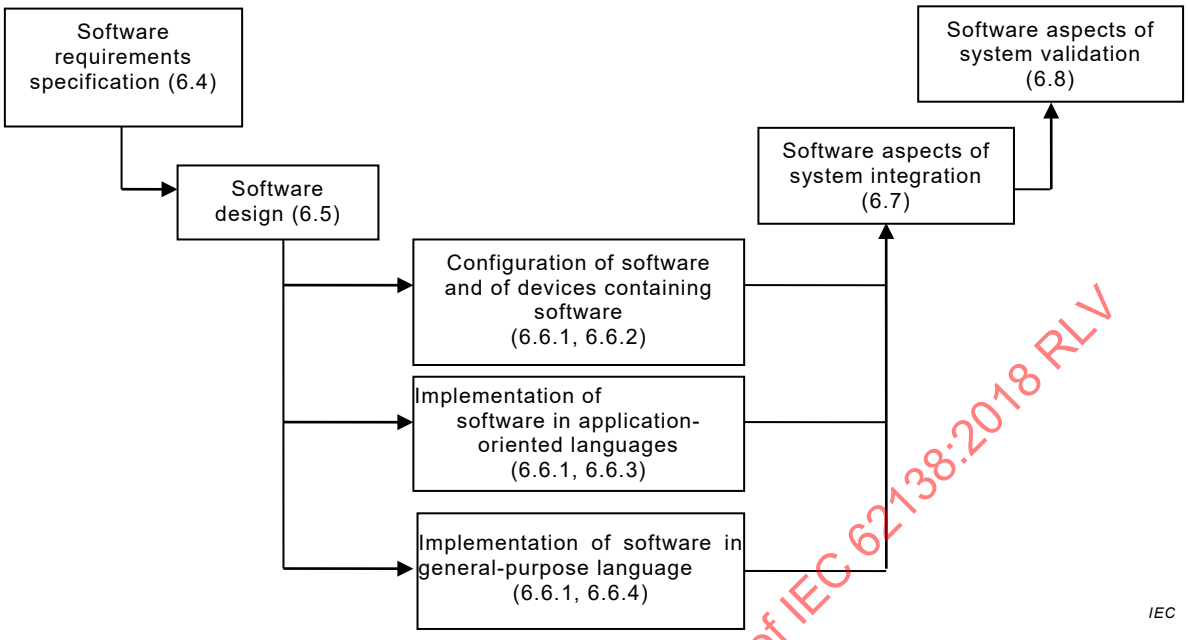


Figure 4 – Development activities of the IEC 62138 software safety lifecycle

5.5 Gradation principles

As a consequence of the gradation of safety relevance for functions of categories A, B and C (see IEC 61226), a suitable gradation has been adopted for the requirements applicable to the software of I&C systems of safety classes 1, 2 and 3.

Software of I&C systems of safety classes 1 is covered by IEC 60880.

The application of the requirements of this document for safety class 3 confers the basic level of confidence that is suitable for software of an I&C system important to safety. The principles followed are:

– reliance on quality assurance;
– special attention given to the assurance that the software:
  • contributes as necessary to, and does not adversely affect, the functions important to safety;
  • satisfies the software requirements specification statements which define constraints important to safety;
– assurance that the operators of the I&C system are informed as early as reasonably possible of software errors and failures that may affect the functions identified as important to safety, so that any appropriate action can be taken;
– documented software requirements specifications, design specifications, integration specifications, validation specifications (i.e. full functional testing) and modification specifications.

For safety class 2, in addition to the principles already stated for class 3, the principles followed by this document are:

– justification, based on tests and design, that the required safety-related performance (for example, response times) will be met in all the specified conditions;
– more stringent requirements for the selection of pre-developed software;

– more stringent requirements for verification, configuration management, selection and use of software tools and languages;

– explicit requirements for simplicity, clarity, precision, verifiability, testability and modifiability.

When requirements are applicable to both safety classes, the extent of the justification required to confirm compliance with this document may be moderated according to the safety class, i.e. for class 3, the extent of justification may be reduced compared to class 2. Also, the extend of justification for those functions which are 'not important to safety' in class 2 or 3 systems need only address how the design ensures that such functions do not jeopardise the functions which are identified as important to safety.

6 Requirements for the software of class 2 and class 3 I&C systems

6.1 Applicability of the requirements

The requirements and recommendations of this document are stated in this Clause 6. The requirements and recommendations that are not specifically marked are applicable to class 2 and class 3 systems. The requirements and recommendations that are applicable specifically to class 3 or to class 2 systems are identified as such and appear in italics.

All the requirements and recommendations are indented and numbered. All other paragraphs are informative. In particular, unnumbered paragraphs provide notes regarding the immediately preceding numbered paragraph unless otherwise stated. When unnumbered paragraphs provide notes regarding more clauses than the immediately preceding numbered paragraph, they are introduced by "Concerning xxx and yyy, ...".

It is not the intention of this document to prescribe a defined set of documents, but rather to define the information which needs to be documented. The particular hierarchy and format of documentation adopted may vary, provided that the principles set out in this document are addressed. For information Annex A presents a typical list of software documentation.

6.2 General requirements

6.2.1 Software safety lifecycle – Software quality assurance

6.2.1.1 General

Subclause 6.3.2 of IEC 61513:2011 provides requirements for quality assurance at the level of an I&C system. This subclause provides additional requirements specific, or of particular importance, to software.

6.2.1.2 The development of software shall be performed according to a software safety lifecycle. The provisions of this software safety lifecycle shall be specified in a quality assurance plan.

This quality assurance plan may be a part of the system quality assurance plan, or may be a separate software quality assurance plan.

6.2.1.3 If a separate software quality assurance plan is used, it shall be consistent with the system quality assurance plan. The software quality assurance plan shall address the requirements of 6.3.2 of IEC 61513:2011 as they relate to software.

6.2.1.4 The quality assurance plan shall divide the development phase of the software safety lifecycle into specified activities. These activities shall include the activities necessary to achieve the required software quality, and to verify and provide objective evidence that this quality is achieved.

6.2.1.5 The specification of an activity shall state:

- its objectives;
- its relationships and interactions with other activities;
- its inputs and results;
- the organisation and responsibilities relevant to the activity.

6.2.1.6 The contents and properties required of the inputs and results should also be specified.

6.2.1.7 The quality assurance plan shall require that the implementation of each activity is assigned to competent persons equipped with adequate resources.

6.2.1.8 The quality assurance plan shall require that modifications in approved documents are identified, reviewed and approved by authorised persons.

6.2.1.9 The quality assurance plan shall require that the methods, languages, tools, rules and standards used are identified and documented, known to, and within the competencies of the concerned development personnel.

6.2.1.10 The quality assurance plan shall require that if several methods, languages, tools, rules and/or standards are used, it is clear which ones have to be used for each activity.

6.2.1.11 The quality assurance plan shall require that project specific terms, expressions, abbreviations and conventions used are explicitly defined.

6.2.1.12 The quality assurance plan shall require that non-conformances raised are tracked and resolved.

6.2.1.13 The quality assurance plan shall require that records resulting from its application are produced. In particular, it shall require that the results of verifications and reviews are recorded together with the scope of the verifications or reviews, the conclusions reached and the resolutions agreed. Any deviation from the quality assurance plan shall be documented and justified.

6.2.1.14 The quality assurance plan shall require that the output documentation constitutes a set of appropriately cross-referenced mutually consistent documents, ensuring the traceability of the final design to the input requirements.

6.2.2 Verification

6.2.2.1 A verification plan shall define the scope of software verification and review activities.

6.2.2.2 The verification plan shall address the requirements of 6.3.2.2 of IEC 61513:2011 as they relate to software.

6.2.2.3 Verifications and reviews shall be performed according to documented provisions. The Verification Plan shall ensure that:

- the verification results are held under configuration management;
- all verification activities have precisely identified inputs, and their results are consistent with these inputs;
- the activities fulfil their specified objectives, and their results have the required contents and properties, and comply with any resolution agreed;
- the results are clear, precise and up-to-date;

- the results comply with any applicable rule;
- the results comply with the applicable requirements of this document.

“Precisely identified” means that the version is known without any ambiguity. “Clear” means that the individuals who need to read a document can fully understand it without excessive effort, even if they have not been involved earlier in the project, provided that they have the required knowledge. “Precise” means that there is no ambiguity.

The extent of the verification and review activities may be dependent on the scale and nature of the software, on the scale and nature of the results to be verified or reviewed, and on the methods and tools used. This extent may also be less thorough regarding the specified requirements that are not identified as important to safety (see 6.4.4.7) and that cannot jeopardise the functions identified as important to safety.

6.2.2.4 The verification plan should ensure that records are produced such that the verification process is fully auditable, i.e. such that independent confirmation of the implementation of the verification plan may be performed.

6.2.2.5 The verification of the results of an activity shall be performed by competent persons who did not participate in the activity.

This does not imply that a person who is an author for one document cannot be the verifier of another.

6.2.2.6 The verification of the results of an activity should include representatives of those concerned with the use of these results, as well as other experts, as necessary.

6.2.2.7 The software requirements specification, the software design specification and the software validation plan shall be verified.

6.2.2.8 *For class 2, the application of design and implementation rules shall be verified*.

6.2.2.9 Software verification shall be performed by persons who did not develop the software being verified.

6.2.2.10 *For class 2, persons who do the verification should have managerial independence from the developers*.

6.2.3 Configuration management

6.2.3.1 General

Subclause 6.3.2.3 of IEC 61513:2011 provides requirements for configuration management at the I&C system level. This subclause provides additional requirements specific, or of particular importance, to software.

6.2.3.2 Configuration management for software shall be performed according to the provisions of a configuration management plan or of the quality assurance plan. These provisions shall be consistent with those for system level configuration management.

6.2.3.3 Configuration management shall be applied to the items related to the correctness of software. The configuration management plan shall specify which software items or types of software items are to be held under configuration management. In particular, these shall include:

- the key documents of the software safety lifecycle (in particular the documents required to be verified);

- the software components necessary to build the executable code, and the executable code itself;
- the software tools influencing the correctness of software.

6.2.3.4 The configuration management plan shall specify technical means for the authentication of the software items under configuration management and of their versions.

6.2.3.5 The configuration management plan shall ensure that the version of the software attached to a given version of the system or equipment, and the versions of the items which together constitute this software version are uniquely identified.

6.2.4 Selection and use of software tools

6.2.4.1 General

Software tools can play an important role in preventing the introduction of faults in software and in revealing existing faults. In particular, tools can aid or automate the design of the architecture of I&C systems and the development of new application software.

6.2.4.2 Software tools should support the development activities which contribute to the correctness of software.

It is usually preferable to focus not only on the quality and on the use of individual tools, but also to consider their compatibility with any other tools to be used, so that together, the tools selected form a coherent tool set. Generally it is preferable to use tools with extensive and relevant operational experience. The use of other tools may be justifiable based upon the requirements of a particular development process.

6.2.4.3 *For class 2, the equipment families used for the development of an I&C system shall be associated with software tools that can reduce the risk of introducing faults in new application software*.

6.2.4.4 *For class 3, the equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in new application software*.

Concerning 6.2.4.3 and 6.2.4.4, these tools usually include support for application-oriented languages, allowing plant and system engineers to specify or verify application functions. Other significant features of such tools may include functional animation, automatic code generation and assistance in the development of functional test specifications.

6.2.4.5 The equipment families used for the development of an I&C system should be associated with software tools that can reduce the risk of introducing faults in the configuration of their pre-developed software and in the design of the system.

Such tools may for example assist system designers in:

- organising the system into a suitable set of interconnected sub-systems;
- distributing the application functions across the sub-systems;
- configuring the sub-systems, their communications and their operational system software;
- ensuring that resources are adequate for all the modes of operation of the system;
- taking into account design and implementation constraints, in particular those aiming at the correctness and robustness of the system.

6.2.4.6 The quality assurance plan shall precisely identify the software tools which may influence the correctness of software.

6.2.4.7 User documentation shall be provided for such tools to ensure that they are used as intended.

6.2.4.8 The quality assurance plan shall distinguish the tools which might introduce faults in software from those which might only lead to overlooking already existing faults.

Code generators and compilers are examples of tools of the first category, whereas static code analysers and test case generators are examples of tools of the second category.

6.2.4.9 *For class 2, the software tools which might introduce faults in software shall be selected and used according to documented procedures and rules aiming at reducing or mitigating this risk. Evidence shall be provided regarding their quality and their ability to produce correct results. Where tools have been applied to generate a given item or information their use shall be recorded to identify them.*

6.2.4.10 *For class 3, evidence should be provided regarding the quality of the software tools which might introduce faults in software and regarding their ability to produce correct results.*

6.2.4.11 Evidence regarding tool quality and ability to produce correct results should be based on operational experience, tool qualification or certification, certification of their suppliers for appropriate development practices, guarantee of appropriate tool development processes, and/or tests. The required stringency of the evidence should be determined based upon the conditions of use of the tool, the extent of the verification of its outputs, the likelihood of tool errors to be detected, and the seriousness of the consequences of undetected erroneous results. Conversely, stringent evidence (for example, a tool qualification according to IEC 60880) may be used as a substitute for some of the verifications of outputs.

6.2.4.12 *For class 2, the software tools which might fail to report faults in software should be selected and used in a way which reduces this risk.*

6.2.4.13 *For class 2, the use of software tools which might fail to report faults in software should be recorded.*

6.2.4.14 *For class 2, when a tool or tool version which has the potential to introduce faults in software is substituted with another, precautions shall be taken to ensure that this does not have adverse effects on the correctness of the software.*

For example, in addition to the quality and ability of the new tool to produce correct results, its compatibility with the previous tool may need to be assessed.

6.2.5 Selection of languages

6.2.5.1 The languages (application-oriented or general-purpose) used to develop software shall have precise and documented syntax and semantics.

6.2.5.2 Application-oriented languages, if available, should be used.

6.2.5.3 *For class 2, low level, machine-oriented general-purpose languages (for example, assembly languages) may be used for specific computer programs, but this should be justified.*

6.2.5.4 When more than one language is used for generating executable code, interfaces between languages shall be documented.

The interface between languages includes argument passing schemes and representation of data structures.

6.2.5.5 *For class 2, the general-purpose languages used should have features facilitating tool supported static analyses of computer programs.*

6.2.5.6 The general-purpose languages used should support explicit and static typing of variables.

6.2.5.7 *For class 2, explicit and static typing of variables should be used.*

6.2.5.8 *For class 2, the languages used and their corresponding run-time libraries shall enable predictable run-time behaviour of the software.*

For example, disruption of the normal behaviour of the software for the collection of freed memory at random moments is usually not acceptable.

6.3 Selection of pre-developed software

6.3.1 General

Subclause 6.2.3.2 of IEC 61513:2011 provides general requirements for the selection of pre-existing components (not necessarily software components). This Subclause 6.3 provides additional requirements specific, or of particular importance, to software.

6.3.1.1 Application software is plant specific and so should not be considered pre-developed software.

NOTE The same application software may be used in multiple units based on the same plant design and safety requirements. In such a case, the justifications produced for the original unit are applicable for the following units.

6.3.2 Documentation for safety

6.3.2.1 Objectives

6.3.2.1.1 Pre-developed software shall have documentation giving the information necessary for using the software safely in the I&C system.

In this document, the corresponding document or set of documents is called documentation for safety. When the pre-developed software is a part of an equipment or equipment family, this documentation may be a part of the documentation for safety of the equipment or equipment family.

Documentation for safety generally comprises more than the user documentation provided by the supplier of the pre-developed software. For example, it may include information obtained from additional tests, measurements and/or analyses, and from operational experience.

6.3.2.2 Contents

6.3.2.2.1 Documentation for safety shall include a description of:

- the functions provided;
- the interfaces with application software;
- the roles, types, formats, ranges and constraints of inputs, outputs, exception signals, parameters and configuration data, where appropriate;
- the different modes of operation and the corresponding conditions of transition;
- any constraint to be respected when using the pre-developed software.

6.3.2.2.2 *For class 2, when applicable, these constraints should:*

- *give adequate confidence in the correctness of the integrated software and of the system design (for example, margins to be taken when using dynamically allocated resources such as memory, processing power, communication bandwidth, operating system resources);*
- *enhance the ability of the integrated software and of the I&C system to detect, signal and tolerate failures, to adopt specified modes of operation and to recover from failures;*
- *give adequate confidence that operator mistakes and failures of other systems or equipment with which the integrated software interacts or shares resources will lead to defined modes of operation;*
- *guarantee that the environment of the pre-developed software will provide all the necessary resources in all conditions of use in the I&C system.*

6.3.2.2.3 When applicable, the documentation for safety should also provide information regarding the performance (for example, in terms of response time) of the functions.

The functions provided by the software, including those related to the system interfaces, may vary depending on the operational conditions of the plant.

6.3.2.2.4 *For class 2, the documentation for safety shall also provide information regarding:*

- *the self-supervision performed, the fault tolerance capability and the failure modes;*
- *the requirements of the pre-developed software regarding its runtime environment (for example, regarding hardware or other software components);*
- *the interactions and interfaces of the pre-developed software with the hardware, to the extent necessary to fully define the safe functional performance of the system.*

6.3.2.2.5 *For class 2, the documentation for safety of the operational system software of a pre-developed equipment family shall provide information enabling (when combined with application-specific data) correct predictions regarding the key safety significant elements of system performance, including notably the maximum response times and the maximum usage of resources.*

Such information may be provided in the form of data, formulae and/or models allowing the calculation of worst case response times and the resource usage of applications. When the software offers a wide range of functions, interfaces and possibilities for configuration, an appropriate confidence in the correctness of the information may be difficult to obtain without knowledge of the operating principles of the software.

6.3.2.3 Properties

6.3.2.3.1 Documentation for safety shall be accurate and shall avoid ambiguity.

6.3.3 Evidence of correctness

6.3.3.1 General requirements

6.3.3.1.1 The correctness of pre-developed software with respect to its documentation for safety shall be justified.

The justification is usually qualitative because there are no generally recognised means to quantify it. Figure 5 and Figure 6 describe a typical process that can be taken. It is recognized however that this is not the only possible approach and that other approaches may be used.

6.3.3.1.2 When using complementary means for providing evidence of correctness, the acceptance criteria should be specified and justified in early stages of the software safety

lifecycle. These criteria should be justified considering the requirements of this document the compliance to which has not been adequately established.

6.3.3.1.3 Pre-developed software should be divided into two different types:

a) Complete operational system software.

b) Software components (real time operating system, library, firmware).

NOTE Application software is plant specific, so it is not considered pre-developed software (see 6.3.1.1).

The rationale behind this distinction is that software components need to be integrated into larger software to form complete operational system software. This means that software components benefit from the development process of the complete operational system software where they are integrated. This allows their functionalities to be verified and validated in the context of their use in the complete operational system software. Therefore the recommended approach to justify the correctness of complete operational system software with respect to its Documentation for Safety (see Figure 5) is more demanding than the recommended approach to justify the correctness of software components (see Figure 6).

6.3.3.1.4 A pre-developed software item should be considered a software component only if it is integrated into larger software to form complete operational system software. Also, a pre-developed software item should be considered a software component only if a later reconfiguration of operational software could not lead to a component being executed in a different way to its initial use (as this would mean that the qualification performed on the complete operational unit would not adequately qualify the software component).

6.3.3.1.5 Verification and validation of the complete operational system software should be performed with the software components embedded. The integration may be within software that runs on a single processor, for example for real time operating systems or libraries. The integration may also be within software that run in close cooperation on several processors, for example the firmware of communication modules or input/output modules.

6.3.3.1.6 *For class 2, the qualification process for software components (see Figure 6) should be used only for pre-developed software components that are non-autonomous executables.*

An 'autonomous executable' is software that can be run by itself without any additional code.

General purpose operating systems designed primarily to be used on workstations are typically autonomous executables in the sense that once they are installed they automatically run many tasks which are not defined by the user.

Libraries (e.g. C library) need to be called by additional code in order to function. A library can be compiled and loaded onto a processor but it will not run unless code has been written to call the functions of the library. A library is therefore not an autonomous executable.

Real time operating systems designed primarily to run embedded software are usually non-autonomous executables in the sense that the user has to define each task explicitly, but this has to be checked on a case by case basis.

6.3.3.1.7 The correctness of a software component with respect to its documentation for safety should be justified by relevant, sufficient and positive operational experience (see 6.3.3.3) or by certification (see 6.3.3.4) (see Figure 6).

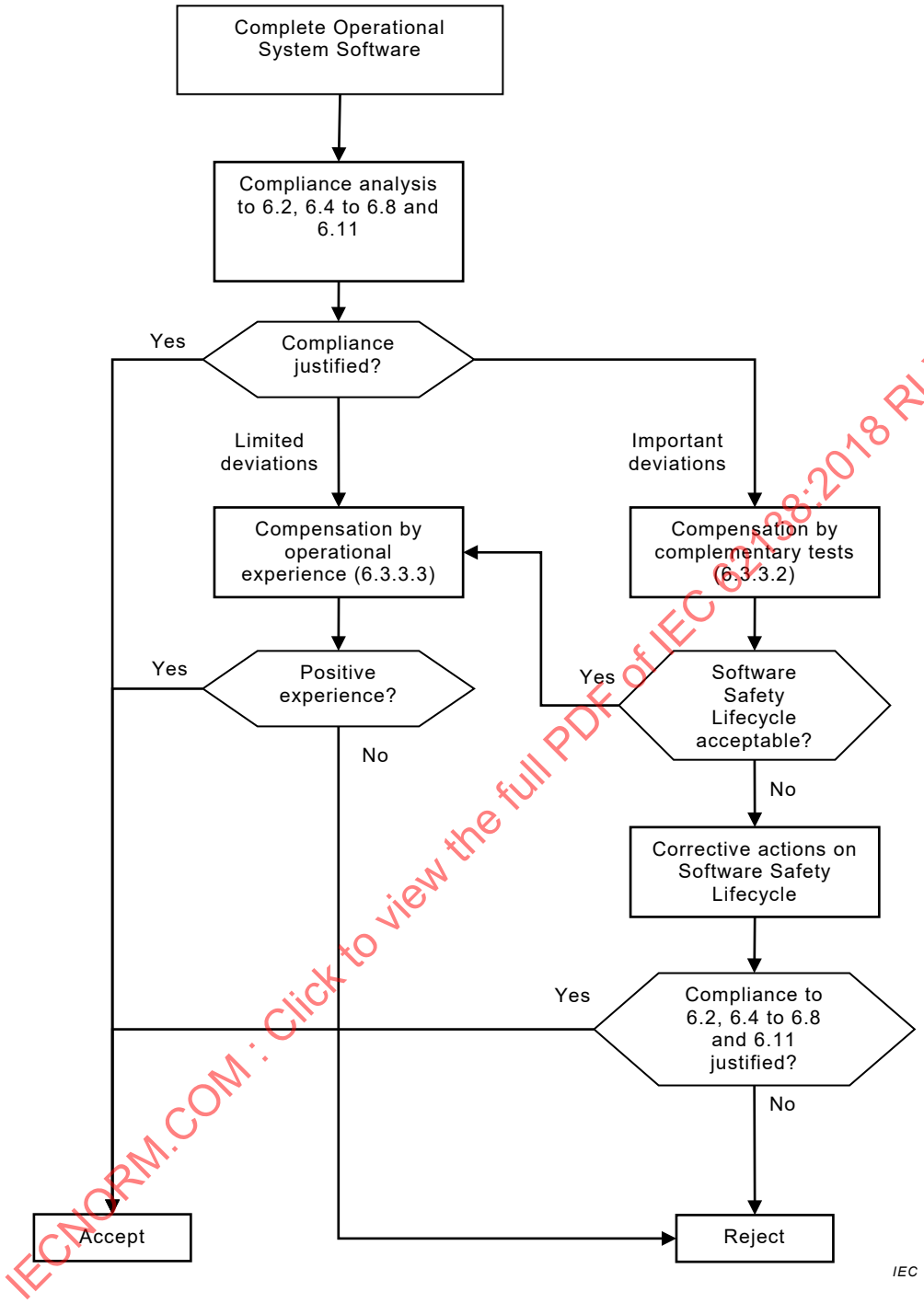


Figure 5 – Overview of the typical qualification process for pre-developed complete operational system software

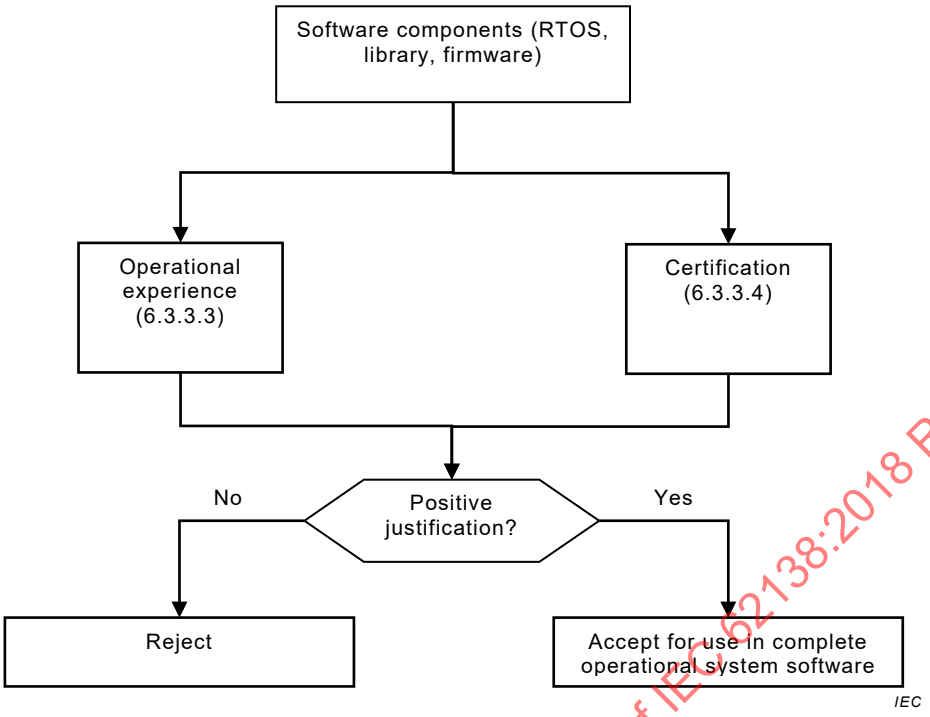


Figure 6 – Overview of the typical qualification process for pre-developed software components

6.3.3.1.8 To justify the correctness of complete operational system software with respect to its documentation for safety, a compliance analysis with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) should be performed first.

6.3.3.1.9 When the compliance analysis shows that the complete operational system software complies with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) then it should be accepted.

When the compliance analysis shows that the complete operational system software has limited deviations with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) then these limited deviations may be compensated by relevant, sufficient and positive operational experience (see 6.3.3.3). In cases where positive operational experience is not available, complementary tests may be used.

Limited deviations are the cases where a full software safety lifecycle has been followed and documented for the complete operational system software but in the execution of the different phases not all the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) have been satisfied.

Concerning for instance the software requirements specification phase (6.4), the case where the software requirements of an I&C system have been specified and documented, but do not include all the contents required in 6.4.4, is an example of a limited deviation.

6.3.3.1.10 When the compliance analysis shows that the complete operational system software has deviations with 6.2.4 then they should be compensated for by relevant, sufficient and positive operational experience (see 6.3.3.3).

6.3.3.1.11 When the compliance analysis shows that the complete operational system software has important deviations with 6.2.2, 6.7 or 6.8, then they should be compensated for by complementary tests (6.3.3.2).

6.3.3.1.12 When the compliance analysis shows that the complete operational system software has important deviations with 6.2.1, 6.2.3, 6.2.5, 6.4, 6.5, 6.6 or 6.11 then the Software safety lifecycle is not acceptable. In such cases, corrective actions should be implemented successfully to accept the software. The goal of the corrective actions should be to achieve compliance with the general subclauses of this document (6.2, 6.4 to 6.6 and 6.11). If corrective actions are not possible, the complete operational system software should be rejected.

Concerning 6.3.3.1.11 and 6.3.3.1.12, important deviations are cases where a full software safety lifecycle has not been followed and documented. The case where a lifecycle has been followed but is not documented is to be interpreted as important deviation.

The case where no validation has been documented is an example of an important deviation.

6.3.3.1.13 The strategy to justify the correctness of pre-developed software with respect to its documentation for safety should be defined and agreed by all parties involved in the early stages of the development of the I&C system.

This strategy cannot be fully defined before the completion of the compliance analysis of the complete operational system software with the general subclauses of this document (6.2, 6.4 to 6.8 and 6.11) as it depends on the gaps that need to be filled.

6.3.3.2 Complementary tests

6.3.3.2.1 General

Complementary tests may be used to support the justification of correctness of pre-developed software, under the following conditions:

6.3.3.2.2 The complementary tests performed on pre-developed software during the development of an I&C system shall be documented.

6.3.3.2.3 The complementary tests shall provide evidence that, in the conditions of use within the I&C system, the pre-developed software is, and behaves as specified by, its documentation for safety.

The conditions of use may concern aspects such as the configuration of the pre-developed software (particularly the setting of parameters and configuration data), the use of functions and interfaces, the hardware environment, the processor and the demand loads.

6.3.3.2.4 *For class 2, the rules used to design complementary tests should be documented and justified.*

6.3.3.2.5 The documentation of complementary tests shall record:

- the version concerned and the configuration of the pre-developed software;
- a description of the tests performed and, when relevant, of the environment used, so as to allow these tests to be repeated in identical conditions;
- the assumptions made to develop the tests, and the evidence of their validity;
- the results obtained, and evidence of their correctness;
- the conclusions reached and the resolutions agreed.

6.3.3.3 Operational experience

6.3.3.3.1 General

Operational experience in systems of a lower safety class, or in non-safety classified systems may be taken into consideration. Operational experience may be used to support the justification of correctness of pre-developed software, under the following conditions:

6.3.3.3.2 The volume of the operational experience taken into consideration shall be documented.

6.3.3.3.3 *For class 2, the operational experience taken into consideration shall correspond to precisely identified versions of the pre-developed software and, when this software is specific to equipment, of the equipment in which it operates.*

6.3.3.3.4 *For class 2, when all or part of the operational experience corresponds to other versions of the pre-developed software and/or of the equipment, the differences with the versions to be used in the I&C system shall be assessed, and the relevance of this operational experience shall be justified.*

6.3.3.3.5 *For class 2, documented justification shall be given that the operational experience taken into consideration corresponds to conditions of use covering those of the I&C system (the intended configuration of the software is one of the conditions of use). In cases where the operational experience conditions are the same, experience in systems of a lower safety class, or in non-safety classified systems can be taken into consideration.*

6.3.3.3.6 *For class 2, the methods used for collecting the operational experience taken into consideration shall be documented. In particular, documented justification shall be given that the failures (if any) caused by the pre-developed software during the operational experience taken into consideration were correctly detected and reported.*

6.3.3.3.7 *For class 2, evidence shall be provided that these failures were correctly analysed, and that the corresponding software faults corrected.*

6.3.3.4 Certification

6.3.3.4.1 General

Pre-developed software used in systems important to safety already in operation (albeit not necessarily in I&C systems of nuclear power plants) may have been certified for compliance to some safety documents. The evidence provided by such a certification may strongly support the justification of correctness of pre-developed software under the following conditions:

6.3.3.4.2 The safety document used for the certification of the pre-developed software shall address explicitly the software development process.

6.3.3.4.3 The certification taken into consideration shall be documented.

6.3.3.4.4 The precise identification of the pre-developed software certified shall be documented. If it was certified as a part of a larger product (for example, as a part of an equipment or equipment family), the precise identification of this product shall also be documented.

6.3.3.4.5 *For class 2, the evidence supporting the certification shall be assessable, in particular:*

- *the conditions (for example, the conditions of use and the assumptions) of the certification;*

- *the methods and tools used for the certification;*
- *the results obtained (for example, the properties and/or measurements certified).*

6.3.3.4.6 *For class 2, the relevance of these conditions and results to the evidence of correctness shall be justified.*

6.3.3.4.7 *For class 2, the effectiveness of the methods and tools used for the certification should be justified.*

6.3.3.4.8 *For class 2, the certifying authority shall be identified and shall be competent for the properties and/or measurements certified.*

6.3.3.4.9 *For class 2, the version of the pre-developed software certified shall be the same as the one used in the I&C system.*

6.3.3.5 Modification

6.3.3.5.1 General

When a well-identified and limited modification is made to pre-developed software for which an appropriate justification of correctness already exists, the following requirements can be used as a substitute for the requirements of 6.3.3.1 to 6.3.3.4 to update or complete the justification. A change in the configuration data of the pre-developed software does not constitute a modification, provided that the new configuration remains within the range covered by the justification.

6.3.3.5.2 The modification of the pre-developed software shall be documented.

6.3.3.5.3 *For class 2, the documentation of the modification shall state:*

- *the precise identification of the modified software;*
- *the context of the modification, if the software is a part of a larger product (for example, an equipment or equipment family);*
- *the objectives, the specification and the constraints of the modification;*
- *the changes made to the documentation for safety.*

6.3.3.5.4 *For class 3, the documentation of the modification should state:*

- *the precise identification of the modified software;*
- *the context of the modification, if the software is a part of a larger product (for example, an equipment or equipment family);*
- *the objectives, the specification and the constraints of the modification;*
- *the changes made to the documentation for safety.*

Concerning 6.3.3.5.3 and 6.3.3.5.4, the context of a modification may for example indicate:

- *the precise identification of the modified larger product;*
- *the objectives, the specification and the constraints of the modification of the product;*
- *the modifications in the rest of the product that need to be made or that may have an impact on the pre-developed software;*
- *the verification and validation actions performed at the level of the product.*

6.3.3.5.5 *For class 2, the documentation should also state the changes made to the design of the pre-developed software.*

6.3.3.5.6 Documented evidence (for example, based on manual inspections, tool supported analyses and/or tests) regarding the modified software, and possibly the larger product, shall justify that:

- the objectives of the modification are satisfied;
- no faults have been introduced;
- the modified software conforms to its updated documentation for safety.

6.3.3.5.7 *For class 2, the sufficiency of this evidence shall be justified, possibly taking into account the modifications made and the conditions of use within the I&C system.*

6.3.3.5.8 The documentation for safety shall be updated as required to maintain its accuracy with respect to any modifications to the software that could affect how the end user installs, operates or maintains the system of which the software is part of.

6.3.4 Functional suitability

6.3.4.1 General

The objective of this subclause is to ensure that pre-developed software is well-suited for the needs of the I&C system, and that it is not too complex with respect to these needs.

6.3.4.2 When applicable, the documentation for safety of pre-developed software shall be evaluated with respect to the system specification and system design. Inconsistencies shall be resolved.

6.3.4.3 *For class 2, the functions of the pre-developed software which are not required to support the system requirements specifications should be identified. A justification that these functions do not have a detrimental effect on safety should be provided*.

6.3.5 Selection and use of digital devices of limited functionality

IEC 62671 may be used as an alternative to this document for digital devices of limited functionality. IEC 62671 contains precise criteria to determine if it is applicable to a particular device.

6.4 Software requirements specification

6.4.1 General

This Subclause 6.4 completes and adds precision to the requirements of 6.2.3.4 of IEC 61513:2011.

6.4.2 Objectives

6.4.2.1 The requirements for the software of an I&C system shall be specified and documented.

The corresponding document or set of documents is called the software requirements specification. In principle, its objective is to specify what the software is to achieve without specifying how it shall do it. However, design and implementation constraints may have to be specified when this is required by considerations of the design of the I&C system or of the I&C architecture.

6.4.2.2 *For class 2, the software requirements specification should avoid unnecessary complexity of the software design*.

6.4.2.3 The software requirements specification shall be such that:

- it contributes to the confidence in the correctness of the design of the I&C system;
- compliance of the I&C system to the requirements of IEC 61513:2011 can be demonstrated.

The IEC 61513:2011 requirements concerned with software requirements specification are mainly in 6.2.2.3, 6.2.2.4, 6.2.2.5, 6.2.3.3, 6.2.3.5 and 6.2.4.

6.4.2.4 The software requirements specification shall be a reference for software design, software validation, and possible software modifications.

6.4.3 Inputs

6.4.3.1 *For class 2, the inputs to the software requirements specification shall include the system specification and the system design documentation.*

They may also include other documents, for example:

- *project specific constraints;*
- *applicable rules and standards;*
- *requirements such as independence between functions;*
- *integrity requirements such as self-supervision to drive outputs to a safe state in the event of detectable failures.*

6.4.3.2 *For class 2, the structure of the software requirements specification should facilitate verification to ensure that it is consistent and complete with respect to its input documents.*

The software requirements specification may reference input documentation directly, so as to avoid unnecessary duplications and minimise the risk of inconsistencies. It may also reference other pre-existing documents, such as the documentation of pre-developed software.

6.4.3.3 The software requirements specification shall provide traceability to its input documents.

6.4.3.4 The verification of the software requirements specification (see 6.2.2) should notably check that it is consistent and complete with respect to its input documents.

6.4.3.5 The references, if any, made by the software requirements specification to other documents shall be precise so as to be unambiguous.

6.4.3.6 *For class 2, the software requirements specification should avoid unnecessary functionality.*

In principle, it is preferable that the software does not have more capabilities than required so as to minimise complexity. However, because current industrial practice is based on the use of pre-developed components, the inclusion of non-required capability may be justified.

6.4.4 Contents

6.4.4.1 The software requirements specification shall specify:

- the application functions to be performed by the software;
- the different modes of operation of the software, and the corresponding conditions of transition;

- the interfaces and interactions of the software with its environment (for example, with operators, with the rest of the I&C system, with the other systems and equipment with which it interacts or shares resources), including the roles, types, formats, ranges and constraints of inputs and outputs;
- the parameters of the software which are to be modified by operators during operation, if any, their roles, types, formats, ranges and constraints, and the checks to be performed by the software when they are modified;
- required performance, when appropriate;
- what the software shall not do or shall avoid, when appropriate;
- the requirements of, or the assumptions to be made by, the software regarding its environment, when applicable.

6.4.4.2 The software requirements specification should also specify the conditions (for example, the demand load), in particular the worst case conditions, provided to the software by its environment.

Concerning 6.4.4.1 and 6.4.4.2, functions, interfaces and performances requirements may depend on the mode of operation, on the values of the parameters, on the configuration data and on the conditions provided to the software.

6.4.4.3 The software requirements specification shall specify the software modes of operation required when errors or failures are detected. When periodic tests are required of the I&C system, the software requirements specification shall also specify the mode of operation required when such tests are performed.

6.4.4.4 The software requirements specification shall state the constraints to be respected by software design and implementation for the sake of correctness and robustness.

For example, this may include constraints:

- to give confidence in the correctness of software and system design (for example, margins to be taken when using dynamically allocated resources such as memory, processing power, communication bandwidth, operating system resources);
- to enhance the ability of the software and of the I&C system to tolerate faults, to detect and signal errors and failures, to take specified modes of operation and to recover from failures;
- to give confidence that operator mistakes and failures of other systems or equipment with which the software interacts or shares resources will not lead to unacceptable effects.

6.4.4.5 The software requirements specification should state the expectations to be respected by software design and implementation for the sake of correctness and robustness.

6.4.4.6 The software requirements specification shall specify the contribution of the software to the assurance that the operators will be informed in due time of errors or failures concerning the functions of the I&C system identified as important to safety. The information provided to the operators shall allow them to take any appropriate action.

6.4.4.7 The software requirements specification shall identify the functions and the requirements related to safety category B or C.

6.4.5 Properties

6.4.5.1 *For class 2, the notations, rules and documents used to develop the software requirements specification should contribute to its clarity and precision, and should be chosen taking into account those used in the inputs and those chosen for the design and implementation of software*.

Since any particular specification format does not always allow a clear, precise and verifiable expression of all specification needs, different and complementary formats may be used in the same software requirements specification. For example, application functions may be specified using a different format than those used for other functions.

6.4.5.2 *For class 2, the requirements of the software requirements specification shall be expressed in such a way that their satisfaction can be assessed objectively*.

6.5 Software design

6.5.1 Objectives

6.5.1.1 The design of software shall be documented.

The corresponding document or set of documents is called the software design specification. When pre-developed software is used, the software design specification may make reference to the corresponding documentation.

6.5.1.2 The software design specification should give an overview of the organisation and of the functioning of the software (see also 6.5.3.3).

6.5.1.3 *For class 2, the software design specification shall contribute to the confidence in the quality of the design of the software, and in its correctness with respect to the software requirements specification*.

6.5.1.4 The software design specification shall provide evidence that the software requirements specification statements important to safety are taken into account in all specified conditions.

6.5.1.5 *For class 2, the software design specification should document the measures taken by the software to ensure that any error or failure of the software is detected early and does not propagate beyond the limits it should specify. It should also document the actions that are taken when an error or failure is detected*.

6.5.1.6 The software design specification shall ensure, if applicable, that the adverse side effects of software errors and failures are cleared prior to returning to a normal mode of operation.

6.5.1.7 The software design shall be produced to achieve modularity, testability and maintainability.

6.5.1.8 *For class 2, providing this does not lead to excessive complexity, the design of the software of an I&C system should facilitate:*

- *the analysis and testing of the software and of its components;*
- *the localisation of faults;*
- *the identification of the effects of a modification.*

6.5.1.9 The software design specification shall be a reference for software implementation and integration, and for possible software modifications.

6.5.2 Inputs

6.5.2.1 The inputs to the software design process shall include the software requirements specification and the documentation for safety of pre-developed software.

They may also include other documents, such as project specific constraints, and/or applicable rules and standards.

6.5.3 Contents

6.5.3.1 The software design specification shall include the specification of:

- the overall organisation of the software;
- the overall functioning of the software under the conditions and modes of operation required by software requirements specification.

6.5.3.2 The overall organisation should provide information regarding:

- the precise identification and the configuration of pre-developed software;
- the distribution of resources, software components and software tasks over sub-systems;
- the allocation of software (sub-)functions to the identified software tasks;
- the main internal interfaces, in particular the interfaces between software tasks.

6.5.3.3 The overall functioning should provide information regarding:

- interactions, communication protocols and information flows;
- sequencing and timing constraints;
- use of resources;
- synchronisation, particularly when using shared resources.

6.5.3.4 *For class 2, the software design specification shall document how the software requirements that are important to safety are met under all specified conditions. When pre-developed software is used, the demonstration regarding the software properties important to safety shall be based in particular on the predictive information provided by the corresponding documentation for safety (see 6.3.2.2.5).*

6.5.3.5 *For class 2, the software design specification and the system design documentation shall state and justify the measures taken to mitigate the effects of the known or anticipated failure modes of any pre-developed software for which complementary means for providing evidence of correctness have been used (see 6.3.3).*

6.5.3.6 *For class 2, the software design specification shall provide rules for software implementation.*

6.5.3.7 *For class 2 the software design specification should in particular specify rules for configuring and for using pre-developed software, so as to ensure that this software is used in a controlled way consistent with the corresponding documentation for safety.*

6.5.3.8 *For class 2, the software design specification shall include the detailed design of any software implemented in general-purpose language.*

6.5.3.9 The software design specification of a component of any software implemented in general-purpose language should specify:

- the functions to be provided by the component, with interfaces, roles, types, formats, ranges and constraints of inputs, outputs, exception signals, and configuration data;
- the required performance (for example, response time, accuracy), when appropriate;
- the requirements of the component regarding its environment (for example, needs in terms of dynamically allocated memory, operating system resources, etc.), when appropriate;
- any other information that the users of the component shall be aware of;
- any relevant implementation constraint.

6.5.3.10 *For class 2, the software design specification shall provide information enabling correct predictions regarding the key safety significant elements of system performance, including notably the maximum response times and the maximum usage of resources.*

Such information may be provided in the form of data, formulae and/or models allowing the calculation of worst case response times and resources usage of applications.

6.5.4 Properties

6.5.4.1 *For class 2, the software design specification shall present the design of the software clearly and precisely.*

6.5.4.2 *For class 3, the software design specification should present the design of the software clearly and precisely.*

Concerning 6.5.4.1 and 6.5.4.2, the main approach may be a top-down approach, but some documents may also give information that highlights how aspects of particular importance (for example, tolerance to failures) are taken into account across the software or across the I&C system.

6.5.4.3 *For class 2, the format and the syntax used to express the design in the software design specification should contribute to clarity and precision.*

6.6 Implementation of software

6.6.1 General requirements

6.6.1.1 General

The requirements of this subclause are applicable to all software, i.e., to the configuration of pre-developed software, and to computer programs written in application-oriented or general–purpose languages.

6.6.1.2 The use of pre-developed software shall be verified to be consistent with the corresponding documentation for safety and with the constraints set by the software design specification.

6.6.1.3 The procedures used to translate computer programs into executable code shall be documented and verified.

These procedures typically describe how the compiler tool chain or the code generator has to be invoked to translate computer programs into executable code. They are often automated.

6.6.1.4 *For class 2, the updating of executable code after changes in computer programs should be performed by automated means.*

6.6.2 Configuration of software and of devices containing software

6.6.2.1 General

The requirement of this subclause is specific to the configuration of customisable software. Such software may be pre-developed or new. However, when the configuration data represents the sequencing of processing to be performed by the software or the system (i.e., it is effectively computer programs), 6.6.3 applies.

6.6.2.2 The configuration of customisable software and devices with embedded customisable software shall be documented.

6.6.3 Implementation with application-oriented languages

6.6.3.1 General

The requirements of this subclause are specific to computer programs written in application-oriented languages. Generally, application oriented formats (such as logic diagrams or function block diagrams) may be used to express all or part of the software requirements specification or of the software design specification. Only limited detailed design and implementation effort is then necessary to transform the specification into computer programs that can be automatically translated into executable code or into a form suitable to be interpreted.

6.6.3.2 The parts of the software requirements specification and/or of the software design specification that are used to generate executable code by automated means shall be considered to be computer programs written in application-oriented languages.

6.6.3.3 *For class 2, computer programs written in application-oriented languages shall be verified to be functionally correct and consistent. The verification shall ensure that:*

- *all the design features are fully understood (i.e., there will be no unexpected behaviour under all specified conditions);*
- *the behaviour specified is consistent with the objectives set by the software design specification.*

Animation, tests, reviews, walkthrough, formal analyses and proof may be applied to improve the understanding of specifications and to verify their functional correctness and consistency.

6.6.3.4 *For class 3, computer programs written in application-oriented languages which are related to functions important to safety shall be verified to be functionally correct and consistent.*

6.6.3.5 The tests shall be developed with respect to the functional requirements of the object under test and not solely to the internal structure of this object.

6.6.3.6 *For class 2, the functional coverage shall be justified prior to the execution of the tests, so that successful execution of the tests confirms the compliance of the object with all its required behaviours.*

6.6.3.7 *For class 2, during test execution, the structural coverage reached by the tests should be monitored with respect to justified criteria (e.g. statement, condition, branch, data flow) in order to ensure the absence of non-required behaviours. Justification should be given if these criteria are not met.*

6.6.3.8 *For class 2, computer programs written in application-oriented languages should conform to documented rules aiming at clarity, modifiability and testability. Non-conformances should be justified.*

A set of rules may be specific to a language or to a set of computer programs. Simplicity, clarity and standardisation of layout and presentation, modularity, presence of relevant comments, avoidance of the unsafe features of the language and of its tools are examples of properties that generally facilitate understanding, verification, testing and later modification.

6.6.4 Implementation with general-purpose languages

6.6.4.1 General

The requirements of this subclause are specific to computer programs written in general-purpose languages.

6.6.4.2 *For class 2, documented verification shall provide evidence that computer programs written in general-purpose languages conform to their specification as defined by the software design specification.*

This may consist of a combination of manual inspections, tool supported analyses, and/or tests.

Code reviews, walkthrough, check lists and other similar techniques are often powerful manual inspection methods that may be considered for identifying software faults.

Tool supported analyses may be used to prove formally that a computer program has (or does not have) given properties. For example, they may give assurance that, under given conditions (for example, that the inputs are within given ranges), the computer program or identified parts of the computer program do not contain certain types of faults (for example, use of non-initialised variables, arithmetic overflow or underflow).

Tests may be performed on the host hardware, or in a software engineering environment.

6.6.4.3 *For class 2, verification documentation shall record:*

- *the identity and the version of the computer programs concerned;*
- *all the information necessary to repeat the verifications in similar conditions;*
- *the assumptions made, and the evidence of their validity;*
- *the results obtained, and evidence of their correctness;*
- *the conclusions reached and, in case of detected errors, the resolutions agreed;*
- *evidence of satisfaction of the acceptance criteria.*

6.6.4.4 The tests shall be developed with respect to the functional requirements of the object under test and not solely to the internal structure of this object.

6.6.4.5 *For class 2, the functional coverage shall be justified prior to the execution of the tests, so that successful execution of the tests confirms the compliance of the object with all its required behaviours.*

6.6.4.6 *For class 2, during test execution, the structural coverage reached by the tests should be monitored with respect to justified criteria (e.g. statement, condition, branch, data flow) in order to ensure the absence of non-required behaviours. Justification should be given if these criteria are not met.*

6.6.4.7 Computer programs written in general-purpose languages shall conform to documented programming rules aiming at clarity, modifiability and testability.

A set of rules may be specific to a language or to a set of computer programs. Simplicity, structured programming, modularity, encapsulation, information hiding (so that users of a software item only have to concern themselves with the service that is provided rather than with the internal workings of the item), presence of relevant comments, avoidance of the unsafe features of the language and of its tools are examples of properties that may facilitate understanding, verification, test and modification.

6.6.4.8 *For class 2, the programming rules should be expressed so as to be verifiable, and should aim in particular at early detection and containment of software errors.*

6.6.4.9 *For class 2, when a static analysis tool can be used to analyse code complexity, then rules should specify acceptable metric limits.*

6.6.4.10 *For class 2, computer programs written in general-purpose languages shall be verified to be compliant with the applicable rules and standards. Non-conformances shall be justified, and appropriate counter-measures shall be taken, documented and justified where necessary.*

For example a counter-measure may be the use of more thorough verification to check that the code is doing what it is intended to do.

6.7 Software aspects of system integration

6.7.1 General

The integration of software is considered as part of system integration. This subclause complements 6.2.5, 6.3.4 and 6.4.5 of IEC 61513:2011 by providing additional requirements specific, or of particular importance, to software.

6.7.2 Software integration and/or inspections shall show that the integrated system and the software:

- comply with the design provisions that ensure the satisfaction of the software requirements specification statements identified as important to safety;
- satisfy the constraints stated by the software requirements specification with respect to correctness and robustness.

6.7.3 *For class 2, when software validation testing has not sufficiently exercised the software, evidence of correct operation of the software shall be obtained, either by performing additional software integration testing or by more thorough verification.*

6.7.4 Software integration shall be performed according to the provisions of the system integration plan or of a software integration plan.

6.7.5 Records of the application of the plan used for software integration shall be produced, for example, test results. In the event of software or system modifications being required, it shall be possible to repeat all, or a subset of, the integration tests to evaluate the extent of possible changes in behaviour.

6.7.6 *For class 2, traceability shall be provided between software design specification and the corresponding integration tests.*

6.7.7 *For class 3, traceability should be provided between software design specification and the corresponding integration tests.*

6.8 Software aspects of system validation

6.8.1 General

Aspects of software functionality are tested during system validation. This subclause complements 6.2.6, 6.3.5 and 6.4.6 of IEC 61513:2011 by providing additional requirements specific, or of particular importance, to software. Where discrepancies are revealed, validation may be continued with justification or may be stopped to correct the discrepancy before revalidation.

6.8.2 *For class 2, software validation shall show that, in the target I&C system, the integrated software conforms to each functional, performance and interface statement of the software requirements specification, and contributes as designed to the satisfaction of the system requirements specification. This shall include justification that:*

- *the specified software functions are correctly performed when their parameters and inputs are in the ranges specified by the software requirements specification, in the conditions of use defined in the software requirements specification;*
- *the system functions to which the software contributes are correctly performed in the conditions of use defined in the system requirements specification;*
- *the software provides defences as required by the software requirements specification against operator mistakes and failures of other systems and equipment;*
- *the software functions as expected in its different modes of operation;*
- *the plant engineering data used by, or integrated in, the I&C system is correct; in particular, the validation of the software shall show that this data defines the interface between the systems and equipment of the plant with which the software interacts or shares resources;*
- *defences required to be performed by the system in the system requirements specification against operator mistakes and failures of other systems and equipment, and to which the software contributes, are correctly provided.*

The validation tests are normally performed with the software integrated in the target I&C system. It may be acceptable to use a platform representative of the target I&C system to perform validation tests if adequate justification is provided.

The conditions of use of functions important to safety may include the concurrent operation of functions not important to safety notably the operation during high communication loading.

6.8.3 *For class 3, software validation shall show that, in the target I&C system, the integrated software conforms to the functional, performance and interface requirements that are identified as important to safety. This shall include justification that:*

- *the specified software functions important to safety are correctly performed when their parameters and inputs are in the ranges specified by the software requirements specification, in the conditions of use defined in the software requirements specification;*
- *the system functions important to safety to which the software contributes are correctly performed in the conditions of use defined in the system requirements specification;*
- *the software provides defences as required by the software requirements specification against operator mistakes and failures of other systems and equipment;*
- *the software functions as expected in its different modes of operation;*
- *the plant engineering data used by, or integrated in, the I&C system to implement functions important to safety is correct; in particular, the validation of the software shall show that this data defines the interface between the systems and equipment of the plant with which the software interacts or shares resources.*

The validation tests are normally performed with the software integrated in the target I&C system. It may be acceptable to use a platform representative of the target I&C system to perform validation tests if adequate justification is provided.

The conditions of use of functions important to safety may include operation during high communication loading.

6.8.4 Software validation shall be performed according to the provisions of a plan that is preferably the system validation plan or a software validation plan.

6.8.5 *For class 2, the plan used for software validation shall specify the validation actions to be performed, and shall show that all the functionality, performance and interface statements of the software requirements specification are correctly taken into account by these actions. It shall also specify the main phases of the software validation (for example, an off-site phase followed by an on-site phase) and the corresponding means, methods and tools to be used.*

6.8.6 *For class 2, the plan used for software validation shall provide traceability between the software requirements specification and the corresponding validation actions.*

6.8.7 *For class 3, the plan used for software validation shall specify the validation actions to be performed, and shall show that all the functionality, performance and interface statements of the software requirements specification identified as important to safety are correctly taken into account by these actions. It shall also specify the main phases of the software validation (for example, an off-site phase followed by an on-site phase) and the corresponding means, methods and tools to be used.*

6.8.8 *For class 3, the plan used for software validation should provide traceability between the software requirements specification and the corresponding validation actions.*

6.8.9 Records of the application of the plan used for software validation shall be produced. In the event of software or system modifications being required, it shall be possible to repeat all, or a subset of, the validation tests to evaluate the extent of possible changes in behaviour.

6.8.10 *For class 2, the results of software validation shall be auditable by persons competent in the subjects addressed but not directly engaged in the validation process.*

6.8.11 *For class 3, the results of software validation should be auditable by persons competent in the subjects addressed but not directly engaged in the validation process.*

6.8.12 These records shall document the configuration of the software being validated and the configuration of the validation environment (for example, the hardware environment and the tools, if any).

6.8.13 The team that writes the plan used for software validation shall include at least one person who did not participate in the design and implementation.

6.9 Installation of software on site

6.9.1 General

Subclause 6.2.7 of IEC 61513:2011 provides requirements regarding the installation of the I&C system on site. This subclause provides additional requirements specific, or of particular importance, to the installation of software.

6.9.2 The procedure for installing software on site shall be documented. It shall guarantee that the correct and complete version of the software is installed.

6.9.3 The procedure for installing software on site shall include and specify on-site checks and tests to be performed before the I&C system is put into full operational use. In particular, the satisfaction of the conditions required for correct operation of the software shall be verified.

For example, these conditions may concern the hardware on which the software operates, or other systems with which the software interacts or shares resources.

6.10 Anomaly reports

6.10.1 If unexpected, apparently incorrect, unexplained or abnormal behaviour is observed after acceptance into service, an anomaly report should be raised.

6.10.2 The anomaly report should give details of the behaviour, the software and hardware configurations and the activities in hand at the time. It should also include the originator, location, date, and a report identification.

6.10.3 The anomaly reports should be reviewed. Issues raised should be documented, tracked and resolved.

6.10.4 The anomaly should be reported to the designer and to the users.

6.11 Software modification

6.11.1 General

The decision to proceed with software modifications depends upon their impact on the I&C system. Therefore, they are subject to the requirements of 6.2.8 and 6.4.7 of IEC 61513:2011. This subclause provides additional requirements specific, or of particular importance, to software.

6.11.2 Software modifications shall be developed and verified so as to maintain consistency with the requirements of 6.2, 6.3, 6.4, 6.5 and 6.6. They shall be installed on-site in accordance with the requirements of 6.9.

6.11.3 Software modifications should be integrated and validated in a manner consistent with 6.7 and 6.8.

6.11.4 When the extent of a modification does not require the full application of 6.7 and 6.8, the integration of the modified software shall be performed according to a regression software integration plan, and the validation shall be performed according to a regression software validation plan. The adequacy and thoroughness of these plans shall be justified taking into account the extent of any modifications made in the software requirements specification and in the software design specification. Records of the application of these plans shall be produced.

6.11.5 *For class 2, when the regression approach is used, the regression software integration plan and the regression software validation plan shall give adequate confidence that the modified software conforms in all respects to the modified software requirements specification, and that:*

- *the objectives of the modification are satisfied;*
- *no fault is introduced;*
- *the modified and/or newly introduced pre-developed software behaves as specified by the corresponding documentation for safety and as expected by the modified software design specification;*
- *the other modified and/or new software components conform to their specification.*

6.11.6 Software modifications shall be comprehensively documented. In particular, all affected software documents shall be updated.

6.11.7 Software modification documentation should state:

- the objectives of the software modification, including any system-level objectives;
- the software components affected or created by the modification;
- identification of the versions of these components, both before and after modification.

The system level objectives of a modification are documented according to the requirements 6.4.7 of IEC 61513:2011.

6.11.8 *For class 2, software modification documentation should state in addition:*

- *any changes made to its specification;*
- *any constraints that need to be respected when developing the modification;*

- *the references of the modified design and/or implementation documents.*

6.11.9 *For class 2, the level of detail of the documentation of a software modification shall be such that:*

- *it contributes as appropriate to the confidence in the correctness of the modified software and I&C system;*
- *compliance of the I&C system to the applicable requirements of IEC 61513:2011 can be demonstrated.*

The IEC 61513:2011 requirements that may be concerned are mainly in 6.2.2.3, 6.2.2.4, 6.2.2.5, 6.2.3.3, 6.2.3.5 and 6.2.4.

6.11.10 The effects of a software modification on the rest of the I&C system and on the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

6.11.11 The effects on software of modifications in the rest of the I&C system or in the other systems with which it interacts or shares resources shall be assessed. Any necessary action shall be taken so as to ensure the correct operation of the I&C system.

6.12 Defences against common cause failure due to software

Systematic faults may be introduced in any design and implementation process due to human error. Therefore such faults may be introduced by errors or omissions in the system/software requirements specification or later during software design and implementation (either in the developed part or in an included pre-existing design). Systematic faults may also be introduced by software tools when such tools suffer themselves from systematic faults introduced in their design and implementation process. Software could therefore potentially be affected by latent systematic faults which could, under some triggering event, lead to the CCF of multiple instantiations of a software design.

The potential for CCF at system level is in the scope of higher level SC 45A Standards, in particular:

- IEC 61513:2011 5.4.2.6 that addresses defence against CCF;
- IEC 61513:2011 5.4.4.2 that addresses the assessment of reliability and defences against CCF.

This document defines development and verification processes and requirements which minimise the potential for software to have systematic faults and therefore, as such faults can cause CCF, also minimise the potential for CCF due to software.

Annex A
(informative)

Typical list of software documentation

Table A.1 gives a typical list of software documentation.

Table A.1 – Typical list of software documentation

References to the subclauses of this document	References
Documents relating to software production	
Software quality assurance plan*	6.2.1
Software verification plan	6.2.2
Software configuration management plan*	6.2.3
Documentation for safety of pre-developed software	6.3
Software requirements specification	6.4
Software design specification	6.5
Programming rules	6.6.3.8, 6.6.4.7
Software verification report	6.2.2, 6.6.3, 6.6.4
Software integration plan*	6.7
Software integration report*	6.7
Software validation plan*	6.8
Software validation report*	6.8
Software installation procedure on site*	6.9
Documents relating to anomaly	
Anomaly report	6.10
Documents relating to software modification	
Software modification documentation	6.11
Regression software integration plan**	6.11
Regression software integration report**	6.11
Regression software validation plan**	6.11
Regression software validation report**	6.11

* These documents may be omitted when their content is included in system documents, for example in the system quality assurance plan, the system configuration management plan, the system integration plan, the system integration report, the system validation plan, the system validation report or the system installation procedure on site.

** When the extent of a modification does not require the full application of 6.7 and 6.8. Subclauses 6.2, 6.3, 6.4, 6.5, 6.6 and 6.9 are always applicable to software modifications and consequently the documents relating to these subclauses have to be kept up to date for any modification.

Annex B
(informative)

Correspondence between IEC 61513:2011 and this document

Table B.1 shows correspondence between IEC 61513:2011 and this document.

Table B.1 – Correspondence between IEC 61513:2011 and this document

IEC 61513:2011 Subclauses		Subclause in this document
5.4.2.5	Tools	6.2.4
5.4.2.6	Defence against CCF	6.12
5.4.4.2	Assessment of reliability and defences against CCF	
5.6.2	Architectural design documentation	6.2.4
6	System safety life cycle, Figure 5	5.4
6.2.2.3.3	Internal behaviour of the system	6.3.2, 6.5
6.2.2.7	Qualification	6.2.4
6.2.3.2	Selection of pre-existing components	6.3
6.2.3.4	Software specification	6.4
6.2.4	System detailed design and implementation	6.5, 6.6
6.2.5	System integration	6.7
6.2.6	System validation	6.8
6.2.7	System installation	6.9
6.2.8	System design modification	6.11
6.3.2	System quality assurance plan	6.2.1
6.3.2.3	System configuration management plan	6.2.3
6.3.4	System integration plan	6.7
6.3.5	System validation plan	6.8
6.4.4	System detailed design documentation	6.5, 6.6
6.4.5	System integration documentation	6.7
6.4.6	System validation documentation	6.8
6.4.7	System modification documentation	6.11
6.5.3.3	Software evaluation and assessment	All
8.2	Requirements on the objectives to be achieved	All

Annex C
(informative)

Relations of this document with IEC 61508

C.1 General

This annex establishes the correspondence between this document and IEC 61508-3:2010.

At the system level, IEC 61513:2011, Annex D establishes the correspondence with IEC 61508-1:2010, IEC 61508-2:2010 and IEC 61508-4:2010.

C.2 Comparison of scope and concepts

IEC 61508 refers to “safety-related systems” in general while this document follows IAEA practice and refers to “systems important to safety” (i.e. important to nuclear safety).

IEC 61508 grades the safety integrity level required for a computer based system according to the risk reduction the system is required to provide. This is arrived at by determining the severity of the risk associated with the hazard, and assessing the frequency and consequences of the hazard and the protection to be provided by the system to reduce the risk from the hazard to a tolerable level.

The nuclear industry has traditionally used primarily a deterministic method to determine the safety significance of a system and its impact on the severity of risk associated with possible discharge of activity.

IEC 61508 requires an independent functional safety assessment by individuals and organisations of experience and independence that rise with the SIL (see Part 1).

In the nuclear sector, the plant operators (who are ultimately responsible for ensuring nuclear safety) are generally in charge of ensuring that adequate functional safety assessment has been performed, but this process is often subject to relevant national nuclear regulations.

C.3 Correspondence between this document and IEC 61508-3:2010

Table C.1 – Correspondence between this document and IEC 61508-3:2010

IEC 62138		IEC 61508-3:2010	
5.4	Software and system safety lifecycles	7.1	General
6.2.1	Software safety lifecycle – Software quality assurance		
6.2.2	Verification	7.9	Software verification
6.2.3	Configuration management	6.2.3	Software configuration management
6.2.4	Selection and use of software tools	7.4.4	Requirements for support tools, including programming languages
6.2.5	Selection of languages		
6.3	Selection of pre-developed software	7.4.2	General requirements
6.3.2	Documentation for safety		Annex D (normative) Safety manual for compliant items – additional requirements for software elements
6.4	Software requirement specification	7.2	Software safety requirements specification
6.5	Software design	7.4	Software design and development
6.6	Implementation of software		
6.7	Software aspects of system integration	7.5	Programmable electronics integration (hardware and software)
6.8	Software aspects of system validation	7.3	Validation plan for software aspects of system safety
		7.7	Software aspects of system safety validation
6.9	Installation of software on site		Outside the scope of IEC 61508-3 as it is addressed in IEC 61508-1
6.10	Anomaly reports		Outside the scope of IEC 61508-3 as it is addressed in IEC 61508-1
6.11	Software modification	7.6	Software operation and modification procedures
		7.8	Software modification
6.12	Defences against common cause failure due to software		IEC 61508-3 addresses defences against common cause failure due to software, in particular in annex C and annex F.
	In the nuclear sector, this assessment is connected to the licensing process and depends on the safety bodies and national regulations.	8	Functional safety assessment IEC 61508-1 imposes requirements for technical knowledge and independence of the functional safety assessor(s) graded on the basis of level of innovation, technical novelty and possible consequences of failures. In addition, most organizations that offer functional safety assessor(s) and product certifications are now accredited by national accrediting agencies.

NOTE Informative annexes of IEC 62138 and IEC 61508 are not considered in Table C.1.

Bibliography

IEC 61508-3:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software requirements*

IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*

IEC 61511-1:2016, *Functional safety – Safety instrumented systems for the process industry sector – Part 1: Framework, definitions, system, hardware and application programming requirements*

IEC 62645:2014, *Nuclear power plants – Instrumentation and control systems – Requirements for security programmes for computer-based systems*

ISO/IEC 12207:2008, *Systems and software engineering – Software life cycle processes*

ISO 9001:2015, *Quality management systems – Requirements*

ISO 90003:2014, *Software engineering – Guidelines for the application of ISO 9001:2008 to computer software*

IAEA Safety Standard Series No. SSR-2/1:2016, *Safety of Nuclear Power Plant: Design*

IAEA Safety Guide SSG-39:2016, *Design of instrumentation and control systems in Nuclear Power Plants*

IAEA Safety Glossary:2016, *Terminology used in nuclear safety and radiation protection*

IAEA Safety Standard Series, N° GS-G-3.5:2009, *the Management System for Nuclear Installations*

IEEE Std 7-4.3.2:2010, *IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations*

DO-178 revision C:2012, *Software Considerations in Airborne Systems and Equipment Certification*

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

CENTRALES NUCLÉAIRES DE PUISSANCE – SYSTÈMES D'INSTRUMENTATION ET DE CONTRÔLE-COMMANDE IMPORTANTS POUR LA SÛRETÉ – ASPECTS LOGICIELS DES SYSTÈMES INFORMATISÉS RÉALISANT DES FONCTIONS DE CATÉGORIE B OU C

AVANT-PROPOS

1) La Commission Electrotechnique Internationale (IEC) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de l'IEC). L'IEC a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, l'IEC – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de l'IEC"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec l'IEC, participent également aux travaux. L'IEC collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de l'IEC concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de l'IEC intéressés sont représentés dans chaque comité d'études.
3) Les Publications de l'IEC se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de l'IEC. Tous les efforts raisonnables sont entrepris afin que l'IEC s'assure de l'exactitude du contenu technique de ses publications; l'IEC ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de l'IEC s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de l'IEC dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de l'IEC et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) L'IEC elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de l'IEC. L'IEC n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à l'IEC, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de l'IEC, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de l'IEC ou de toute autre Publication de l'IEC, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de l'IEC peuvent faire l'objet de droits de brevet. L'IEC ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

La Norme internationale IEC 62138 a été établie par le sous-comité 45A: Systèmes d'instrumentation, de contrôle-commande et d'alimentation électrique des installations nucléaires, du comité d'études 45 de l'IEC: Instrumentation nucléaire.

Cette deuxième édition annule et remplace la première édition publiée en 2004. Cette édition constitue une révision technique.

Cette édition inclut les modifications techniques majeures suivantes par rapport à l'édition précédente:

a) aligner la présente norme sur les normes publiées ou révisées depuis sa première édition, en particulier l'IEC 61513, l'IEC 60880, l'IEC 62645 et l'IEC 62671;

b) fusionner les Articles 5 et 6 de la première édition en un seul article pour éviter la répétition d'une grande partie du texte dont le maintien de la cohérence s'est avéré très difficile;

c) réviser l'article portant sur la sélection des logiciels prédéveloppés sur la base du retour d'expérience issu de l'application de la première édition de la norme pour des projets industriels. Des critères plus précis sont proposés pour ce qui concerne la preuve de conformité des logiciels prédéveloppés;

d) introduire des exigences portant sur la traçabilité en cohérence avec l'IEC 61513;

e) introduire une Annexe A fournissant une liste typique de documentation des logiciels;

f) introduire une Annexe B établissant les relations liant l'IEC 61513 au présent document;

g) introduire une Annexe C établissant les relations liant l'IEC 61508 au présent document.

Le texte de cette Norme internationale est issu des documents suivants:

FDIS	Rapport de vote
45A/1201/FDIS	45A/1209/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette Norme internationale.

Ce document a été rédigé selon les Directives ISO/IEC, Partie 2.

Dans ce document, les caractères suivant sont utilisés:

- *Les exigences et recommandations applicables uniquement aux systèmes de classes 2 et 3 apparaissent en italique à l'Article 6.*

Le comité a décidé que le contenu de ce document ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous "http://webstore.iec.ch" dans les données relatives au document recherché. A cette date, le document sera

- reconduit,
- supprimé,
- remplacé par une édition révisée, ou
- amendé.

INTRODUCTION

a) Contexte technique, questions importantes et structure du présent document

La présente norme internationale établit des exigences portant sur les aspects logiciels des systèmes d'instrumentation et de contôle-commande (I&C) informatisés réalisant des fonctions de catégorie B ou C, telles que définies par l'IEC 61226. Elle complète l'IEC 60880 qui établit les exigences pour les logiciels des systèmes d'I&C informatisés réalisant des fonctions de catégorie A.

Elle est cohérente et complémentaire avec l'IEC 61513:2011. Les activités se situant principalement au niveau système (par exemple l'intégration, la validation et l'installation) ne sont pas couvertes de façon exhaustive par le présent document; les exigences qui ne sont pas spécifiques au logiciel sont reportées dans l'IEC 61513:2011.

Le présent document prend en compte les pratiques de développement actuellement mises en œuvre pour les logiciels de systèmes d'I&C, et en particulier:

– l'utilisation de logiciels, d'équipements et de familles d'équipements prédéveloppés qui n'ont pas nécessairement été conçus selon les normes de l'industrie nucléaire;

– l'utilisation de langages orientés application.

b) Position du présent document dans la collection de normes du SC 45A de l'IEC

L'IEC 61513 est le document de premier niveau du SC 45A qui fournit les recommandations applicables pour l'I&C au niveau système.

L'IEC 62138 est le document de deuxième niveau du SC 45A qui complète l'IEC 61513 pour ce qui est du développement logiciel pour les systèmes d'I&C informatisés réalisant des fonctions de catégorie B ou C.

Pour plus de détails sur la structure de la collection de normes du SC 45A de l'IEC, voir le point d) de cette introduction.

c) Recommandations et limites relatives à l'application du présent document

Il n'est pas prévu que le présent document soit utilisé comme un guide de génie logiciel généraliste. Elle est applicable pour les logiciels des systèmes d'I&C réalisant des fonctions de catégorie B ou C pour les nouvelles centrales nucléaires de puissance comme pour les mises à jour ou les rénovations d'I&C de centrales existantes.

Pour les centrales existantes, seul un sous ensemble d'exigences est applicable et ce sous ensemble a à être identifié au début de chaque projet.

L'objectif des recommandations fournies par le présent document est de réduire, autant que faire se peut, le potentiel de défauts logiciels latents pouvant causer des défaillances système, dues à des défaillances logicielles uniques ou bien à des défaillances logicielles multiples (c'est-à-dire Défaillances de Cause Commune dues au logiciel).

Le présent document ne traite pas explicitement de la protection des logiciels contre les menaces liées à des attaques malveillantes des systèmes informatisés, c'est-à-dire de cybersécurité. L'IEC 62645 fournit des exigences portant sur les programmes de sécurité applicables pour les systèmes informatisés.

Afin d'assurer la pertinence du présent document pour les années à venir, l'accent est mis sur les questions de principes plutôt que sur les technologies particulières.

d) Description de la structure de la collection des normes du SC 45A de l'IEC et relations avec d'autres documents de l'IEC, et d'autres organisations (AIEA, ISO)

Les documents de niveau supérieur de la collection de normes produites par le SC 45A de l'IEC sont les normes IEC 61513 et IEC 63046. L'IEC 61513 traite des exigences générales relatives aux systèmes et équipements d'instrumentation et de contrôle-commande (systèmes d'I&C) utilisés pour accomplir les fonctions importantes pour la sûreté des centrales nucléaires. L'IEC 63046 traite des exigences générales relatives aux systèmes d'alimentation électrique; elle couvre les systèmes d'alimentation électrique jusqu'à et y compris les alimentations des systèmes d'I&C. Les normes IEC 61513 et IEC 63046 doivent être considérées ensemble et au même niveau. Les normes IEC 61513 et IEC 63046 structurent la collection de normes du SC 45A de l'IEC et forment un cadre

complet, cohérent et consistant établissant les exigences générales relatives aux systèmes d'I&C et électriques des centrales nucléaires de puissance.

Les normes IEC 61513 et IEC 63046 font directement référence aux autres normes du SC 45A de l'IEC traitant de sujets génériques, tels que la catégorisation des fonctions et le classement des systèmes, la qualification, la séparation des systèmes, la défense contre les défaillances de cause commune, la conception des salles de commande, compatibilité électromagnétique, la cybersécurité, les aspects logiciels et matériels relatifs aux systèmes numériques programmables, la coordination des exigences de sûreté et de sécurité et la gestion du vieillissement. Il convient de considérer que ces normes, de second niveau, forment, avec les normes IEC 61513 et IEC 63046, un ensemble documentaire cohérent.

Au troisième niveau, les normes du SC 45A de l'IEC, qui ne sont généralement pas référencées directement par les normes IEC 61513 ou IEC 63046, sont relatives à des matériels particuliers, à des méthodes ou à des activités spécifiques. Généralement ces documents, qui font référence aux documents de deuxième niveau pour les activités génériques, peuvent être utilisés de façon isolée.

Un quatrième niveau qui est une extension de la collection de normes du SC 45A de l'IEC correspond aux rapports techniques qui ne sont pas des documents normatifs.

Les normes de la collection produite par le SC 45A de l'IEC sont élaborées de façon à être en accord avec les principes de sûreté et de sécurité de haut niveau établis par les normes de sûreté de l'AIEA pertinentes pour les centrales nucléaires, ainsi qu'avec les documents pertinents de la collection de l'AIEA pour la sécurité nucléaire (NSS), en particulier avec le document d'exigences SSR-2/1 qui établit les exigences de sûreté relatives à la conception des centrales nucléaires, avec le guide de sûreté SSG-30 qui traite du classement de sûreté des structures, systèmes et composants des centrales nucléaires, avec le guide de sûreté SSG-39 qui traite de la conception de l'instrumentation et du contrôle-commande des centrales nucléaires, avec le guide de sûreté SSG-34 qui traite de la conception des systèmes d'alimentation électrique des centrales nucléaires, et avec le guide de mise en œuvre NSS17 traitant de la sécurité informatique pour les installations nucléaires. La terminologie et les définitions utilisées pour la sûreté et la sécurité dans les normes produites par le SC 45A sont conformes à celles utilisées par l'AIEA.

Les normes IEC 61513 et IEC 63046 ont adopté une présentation similaire à celle de l'IEC 61508, avec un cycle de vie d'ensemble et un cycle de vie des systèmes. Au niveau sûreté nucléaire, les normes IEC 61513 et IEC 63046 sont l'interprétation des exigences générales de l'IEC 61508-1, de l'IEC 61508-2 et de l'IEC 61508-4 pour le secteur nucléaire. Dans ce domaine, l'IEC 60880, l'IEC 62138 et l'IEC 62566 correspondent à l'IEC 61508-3 pour le secteur nucléaire. Les normes IEC 61513 et IEC 63046 font référence aux normes ISO ainsi qu'aux documents AIEA GS-R-3 et AIEA GS-G-3.1 et AIEA GS-G-3.5 pour ce qui concerne l'assurance qualité. Au second niveau, l'IEC 62645 est le document chapeau des normes du SC 45A de l'IEC portant sur la sécurité nucléaire. Elle est élaborée sur les principes pertinents de haut niveau de l'ISO/IEC 27001 et l'ISO/IEC 27002; elle les adapte et les complète pour qu'ils deviennent pertinents pour le secteur nucléaire; elle est coordonnée étroitement avec l'IEC 62443. Au second niveau, l'IEC 60964 est le document chapeau des normes du SC 45A de l'IEC portant sur les salles de commande et l'IEC 62342 est le document chapeau des normes du SC 45A de l'IEC portant sur la gestion du vieillissement.

NOTE 1 Il est fait l'hypothèse que pour la conception des systèmes d'I&C qui sont supports de fonctions de sûreté conventionnelle (par exemple pour garantir la sécurité des travailleurs, la protection des biens, la prévention contre les risques chimiques, la prévention contre les risques liés au procédé énergétique) on applique des normes nationales ou internationales.

NOTE 2 Le domaine du SC 45A de l'IEC a été étendu en 2013 pour couvrir les systèmes électriques. En 2014 et en 2015 des discussions ont eu lieu au sein du SC 45A de l'IEC pour décider de la façon et de l'endroit pour établir les exigences générales portant sur la conception des systèmes électriques. Les experts du SC 45A de l'IEC ont recommandé que pour établir des exigences générales pour les systèmes électriques une norme indépendante soit développée au même niveau que l'IEC 61513. Le projet IEC 63046 est lancé pour atteindre cet objectif. Lorsque l'IEC 63046 sera publiée, la présente NOTE 2 de l'introduction sera supprimée.

CENTRALES NUCLÉAIRES DE PUISSANCE – SYSTÈMES D'INSTRUMENTATION ET DE CONTRÔLE-COMMANDE IMPORTANTS POUR LA SÛRETÉ – ASPECTS LOGICIELS DES SYSTÈMES INFORMATISÉS RÉALISANT DES FONCTIONS DE CATÉGORIE B OU C

1 Domaine d'application

Le présent document spécifie des exigences sur les logiciels des systèmes d'instrumentation et de contrôle-commande (I&C) informatisés réalisant des fonctions de sûreté de catégorie B ou C, selon la définition donnée par l'IEC 61226. Il est complémentaire à l'IEC 60880 qui énonce des exigences sur le logiciel des systèmes d'I&C informatisés réalisant des fonctions de sûreté de catégorie A.

Il est également cohérent et complémentaire à l'IEC 61513. Les activités de nature essentiellement système (par exemple l'intégration, la validation et l'installation sur site) n'y sont pas traitées exhaustivement: les exigences qui ne sont pas spécifiques au logiciel sont reportées dans l'IEC 61513.

La relation entre les catégories des fonctions et les classes des systèmes est fournie par l'IEC 61513. Un système d'I&C classé de sûreté pouvant réaliser des fonctions de catégories différentes, ainsi que des fonctions non classées, les exigences du présent document sont attachées à la classe de sûreté du système d'I&C (classe 2 ou classe 3).

Il n'est pas prévu que le présent document soit utilisé comme un guide de génie logiciel généraliste. Il est applicable pour les logiciels des systèmes d'I&C de classe de sûreté 2 ou 3 pour les nouvelles centrales nucléaires de puissance comme pour les mises à jour ou les rénovations d'I&C de centrales existantes.

Pour les centrales existantes, seul un sous ensemble d'exigences est applicable et ce sous ensemble a à être identifié au début de chaque projet.

L'objectif des recommandations fournies par le présent document est de réduire, autant que faire se peut, le potentiel d'avoir des défauts logiciels latents pouvant causer des défaillances système, due à des défaillances logicielles uniques ou bien à des défaillances logicielles multiples (c'est-à-dire Défaillances de Cause Commune dues au logiciel).

Le présent document ne traite pas explicitement de la protection des logiciels contre les menaces liées à des attaques malveillantes des systèmes informatisés, c'est-à-dire de cybersécurité. L'IEC 62645 fournit des exigences portant sur les programmes de sécurité applicables pour les systèmes informatisés.

2 Références normatives

Les documents suivants cités dans le texte constituent, pour tout ou partie de leur contenu, des exigences du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC 60880:2006, *Centrales nucléaires de puissance – Instrumentation et contrôle-commande importants pour la sûreté – Aspects logiciels des systèmes programmés réalisant des fonctions de catégorie A*

IEC 61226, *Centrales nucléaires de puissance – Instrumentation et contrôle-commande importants pour la sûreté – Classement des fonctions d'instrumentation et de contrôle-commande*

IEC 61513:2011, *Centrales nucléaires de puissance – Instrumentation et contrôle-commande importants pour la sûreté – Exigences générales pour les systèmes*

IEC 62671:2013, *Centrales nucléaires de puissance – Instrumentation et contrôle-commande importants pour la sûreté – Sélection et utilisation des appareils numériques à fonctionnalités limitées*

3 Termes et définitions

Pour les besoins du présent document les termes et définitions qui suivent s'appliquent.

L'ISO et l'IEC tiennent à jour des bases de données terminologiques destinées à être utilisées en normalisation, consultables aux adresses suivantes:

- IEC Electropedia: disponible à l'adresse http://www.electropedia.org/
- ISO Online browsing platform: disponible à l'adresse http://www.iso.org/obp

3.1
animation
processus par lequel le comportement défini par une spécification est visualisé avec ses valeurs effectives dérivées des équations de comportement et des valeurs d'entrée

[SOURCE: IEC 60880:2006, 3.1]

3.2
fonction d'application
fonction d'un système d'I&C qui accomplit une tâche relative au processus sous contrôle plutôt qu'au fonctionnement du système lui-même

[SOURCE: IEC 61513:2011, 3.1]

3.3
logiciel d'application
partie du logiciel d'un système d'I&C qui exécute les fonctions d'application

Note 1 à l'article: Le logiciel d'application est à mettre en regard avec le «logiciel système».

Note 2 à l'article: Les logiciels d'application sont propres à la centrale ainsi ils ne peuvent pas être considérés comme des logiciels prédéveloppés.

[SOURCE: IEC 61513:2011, 3.2, modifié (notes à l'article modifiées)]

3.4
langage orienté application
langage informatique spécifiquement conçu pour un certain type d'application et pour être utilisé par les spécialistes de ce type d'application

Note 1 à l'article: Les familles d'équipements offrent en général des langages orientés application de façon à faciliter l'adaptation des équipements à des exigences particulières.

Note 2 à l'article: Les langages orientés application peuvent être utilisés pour la spécification d'exigences fonctionnelles que doit satisfaire un système d'I&C, et / ou pour spécifier ou concevoir le logiciel d'application. Ils peuvent être basés sur du texte, des diagrammes ou une combinaison des deux.

Note 3 à l'article: Exemples: les langages à blocs fonctionnels, les langages définis par l'IEC 61131-3.

Note 4 à l'article: Voir aussi “langage généraliste”.

[SOURCE: IEC 60880:2006, 3.3, modifié (note 4 à l’article ajoutée)]

3.5
défaillance de cause commune
DCC
défaillance de plusieurs structures, systèmes ou composants due à un évènement ou à une cause spécifique unique

Note 1 à l'article: Les causes communes peuvent être internes ou externes au système d’I&C.

[SOURCE: Glossaire de sûreté de l'AIEA, édition 2016]

3.6
complexité
degré de difficulté à comprendre ou vérifier la conception, la mise en œuvre ou le comportement d'un système ou d'un composant

[SOURCE: IEC 61513:2011, 3.9]

3.7
programme <d’ordinateur>
ensemble ordonné d’instructions et de données qui spécifie des opérations sous une forme adaptée à l’exécution par un ordinateur

Note 1 à l'article: Ceci inclut les programmes classiques écrits en langage généraliste et cela inclut aussi les programmes écrits en langage orienté application.

[SOURCE: IEC 60880:2006, 3.10, modifié (note 1 à l’article ajoutée)]

3.8
élément informatisé
élément qui s'appuie sur des instructions logicielles s'exécutant sur des microprocesseurs ou des microcontrôleurs

Note 1 à l'article: Dans ce terme et sa définition le mot «élément» peut être remplacé par les mots «système», «équipement» ou «dispositif».

Note 2 à l'article: Un élément informatisé est un type d'élément numérique programmable.

Note 3 à l'article: Ce terme équivaut au terme «élément programmé».

Note 4 à l'article: Dans la traduction française des normes du SC 45A, les termes «informatique» et «informatisé» sont équivalents.

3.9
gestion de la configuration
processus consistant à identifier et à consigner les caractéristiques des structures, systèmes et composants (y compris des systèmes informatisés et des logiciels) d'une installation, et à s'assurer que les modifications de ces caractéristiques sont correctement élaborées, évaluées, approuvées, publiées, mises en œuvre, vérifiées, enregistrées et incorporées dans la documentation relative à cette installation

[SOURCE: Glossaire de sûreté de l'AIEA, édition 2016]

3.10
cybersécurité
ensemble des activités et des mesures dont l'objectif est d'empêcher, de détecter et de réagir aux attaques digitales dont l'intention est d'entraîner:

– la divulgation d'informations qui pourraient être utilisées pour réaliser des actes malveillants qui pourraient amener à un accident, une situation non sûre ou dégrader les performances de fonctionnement de la centrale (confidentialité),
– les modifications malveillantes de fonctions qui pourraient porter atteinte à la fourniture ou à l'intégrité d'un service demandé par des systèmes programmés-HPD d'I&C (y compris la perte de contrôle) qui pourraient avoir pour conséquence un accident, l'apparition d'une situation non sûre ou une dégradation des performances de l'installation (intégrité),
– la rétention, la prévention pour l'accès à ou la communication d'informations, de données ou de ressources (y compris la perte de vue) malveillantes qui pourraient compromettre la fourniture par un système d'I&C d'un service demandé qui pourrait avoir pour conséquence un accident, l'apparition d'une situation non sûre ou une dégradation des performances de l'installation (disponibilité).

Note 1 à l'article: Cette définition est taillée sur mesure par rapport au domaine de l'IEC 62645, se concentrant sur la prévention, la détection et la réaction aux actes malveillants portant atteinte aux systèmes programmés-HPD d'I&C en utilisant des moyens numériques. Il est reconnu que le terme " cybersécurité " a un sens plus large au niveau des autres normes et documents guide et souvent qu'il couvre les menaces non malveillantes, les erreurs humaines et la protection contre les risques naturels, qui sont en dehors du domaine de l'IEC 62645.

[SOURCE: IEC 62645:2014, 3.6, modifié (suppression de la note 2 à l'article)]

3.11
fonctionnalité dédiée
propriété des appareils qui ont été conçus pour réaliser seulement une fonction clairement définie ou bien à un ensemble très réduit de fonctions, telles que par exemple, la capture et l'envoi d'un paramètre procédé, ou la transformation d'une source de courant alternatif en courant continu. Cette fonction (ou cet ensemble réduit de fonctions) est inhérent à l'appareil, et n'est pas le résultat d'une programmation par l'utilisateur

Note 1 à l'article: Les fonctions auxiliaires (par exemple, l'auto-surveillance, l'auto-étalonnage, la communication de données) peuvent aussi être réalisées dans l'appareil, mais pour autant cela ne change pas le domaine étroit d'application de l'appareil.

Note 2 à l'article: «Dédiés» dans le sens utilisé dans l'IEC 62671 fait référence à une conception pour une fonction particulière qui ne peut pas être modifiée sur le terrain.

[SOURCE: IEC 62671:2013, 3.7]

3.12
spécification de conception
document ou ensemble de documents qui décrivent l'organisation et le fonctionnement d'un élément, et qui sont utilisés comme base de la mise en œuvre et pour l'intégration de l'élément

3.13
documentation de sûreté
document ou ensemble de documents qui spécifient comment un produit peut être utilisé de façon sûre dans une application importante pour la sûreté

Note 1 à l'article: Cette définition est utilisée dans le contexte des logiciels prédéveloppés (voir 6.3).

3.14
analyse dynamique
processus consistant à évaluer un système ou un composant sur la base de son comportement pendant l'exécution. S'oppose à l'analyse statique

[SOURCE: IEC 60880:2006, 3.15]

3.15
élément électrique/électronique/électronique programmable
élément E/E/PE
élément réalisé à base de technologie électrique (E) et/ou électronique (E) et/ou électronique programmable (PE)

Note 1 à l'article: Dans ce terme et sa définition le mot "élément" peut être remplacé par les mots «système», «équipement» ou «dispositif».

[SOURCE: IEC 61508-4:2010, 3.2.13, modifié ("élément" ajouté et note à l'article modifiée)]

3.16
famille d'équipements
ensemble de composants matériels et logiciels pouvant travailler de manière complémentaire dans une ou plusieurs architectures définies (configurations). Le développement des configurations spécifiques à la centrale et du logiciel d'application associé peut être réalisé par des outils logiciels. Une famille d'équipements fournit normalement un certain nombre de fonctionnalités standard (bibliothèque des fonctions d'application) qui peuvent être combinées pour générer un logiciel d'application spécifique

Note 1 à l'article: Une famille d'équipements peut être un produit provenant d'un fabricant ou un ensemble de produits interconnectés et adaptés par un fournisseur.

Note 2 à l'article: Le terme «plate-forme de composants» est parfois utilisé comme synonyme de «famille d'équipements».

[SOURCE: IEC 61513:2011, 3.17, modifié (suppression de la note 1 à l'article)]

3.17
erreur
différence entre une valeur ou condition calculée, observée ou mesurée et la valeur ou condition réelle, spécifiée ou théorique

Note 1 à l'article: Voir aussi «erreur humaine», «défaut», «défaillance».

[SOURCE: IEC 61513:2011, 3.18, modifié (note 1 à l'article ajoutée)]

3.18
code exécutable
logiciel présent dans le système cible

Note 1 à l'article: Le code exécutable comprend généralement les instructions devant être exécutées par le matériel du système cible et les données associées.

3.19
défaillance
incapacité d'une structure, d'un système ou d'un composant de fonctionner conformément aux critères d'acceptation

Note 1 à l'article: L'équipement est considéré comme défaillant lorsqu'ils ne fonctionnent plus, que l'on en ait besoin ou non à ce moment-là. Par exemple, la défaillance d'un système de secours peut ne pas être manifeste jusqu'à ce que l'on ait recours à ce système, soit dans le cadre d'essais, soit lorsque le système principal est en panne.

Note 2 à l'article: Une défaillance est le résultat d'un défaut du matériel, d'un défaut du logiciel, d'un défaut du système ou d'une erreur de maintenance. Elle est engendrée par la trajectoire du signal associé.

Note 3 à l'article: Voir aussi «erreur humaine», «défaut», «erreur».

[SOURCE: Glossaire de sûreté de l'AIEA, édition 2016]

3.20
défaut
imperfection dans un composant matériel, logiciel ou système

Note 1 à l'article: Les défauts peuvent provenir de défauts aléatoires, par exemple suite au vieillissement du matériel, et peuvent être systématiques, par exemple des défauts logiciels, suite à des erreurs de conception.

Note 2 à l'article: Un défaut (notamment un défaut de conception) peut ne pas être détecté dans le système jusqu'à l'apparition d'une situation pour laquelle le résultat produit n'est pas conforme à ce qui était prévu pour la fonction, c'est-à-dire qu'une défaillance se produit.

Note 3 à l'article: Voir aussi «erreur humaine», «erreur», «défaillance».

[SOURCE: IEC 61513:2011, 3.21, modifié (note 3 à l'article modifiée)]

3.21
microprogramme
logiciel étroitement dépendant des caractéristiques du matériel sur lequel celui ci est installé. La présence de microprogramme est généralement «transparente» pour l'utilisateur du composant matériel et ainsi il peut être effectivement considéré comme faisant partie intégrante de la conception du matériel (un bon exemple est le microcode d'un processeur). Généralement, le microprogramme ne peut être modifié par un utilisateur qu'en remplaçant le composant matériel (par exemple puce du processeur, carte, EPROM) qui contient ce logiciel par des composants contenant le logiciel modifié, et lorsque c'est le cas, la gestion de configuration des composants matériels assure effectivement la gestion de configuration des microprogrammes. Le microprogramme, tel que considéré dans l'IEC 60987 est effectivement du logiciel «embarqué» sur le matériel

[SOURCE: IEC 60987:2007, 3.4]

3.22
validation fonctionnelle
vérification de la conformité des spécifications des fonctions d'application aux exigences des fonctions et des performances de haut niveau de la centrale. Elle est complémentaire de la validation du système, qui vérifie la conformité du système à la spécification des fonctions

[SOURCE: IEC 61513:2011, 3.23]

3.23
langage généraliste
langage informatique conçu pour s'adresser à tout type de besoin

Note 1 à l'article: Le logiciel système d'une famille d'équipements est en général réalisé à l'aide de langages généralistes.

Note 2 à l'article: Par exemple, Ada, C, Pascal.

Note 3 à l'article: Voir aussi «langage orienté application».

[SOURCE: IEC 60880:2006, 3.20, modifié (note 3 à l'article ajoutée)]

3.24
erreur (ou faute) humaine
action humaine conduisant à un résultat indésirable

Note 1 à l'article: Voir aussi «défaut», «erreur», défaillance».

[SOURCE: IEC 61513:2011, 3.26, modifié (note 1 à l'article ajoutée)]

3.25
architecture de l'I&C
structure organisant les systèmes de CC de la centrale importants pour la sûreté

[SOURCE: IEC 61513:2011, 3.27]

3.26
système d'I&C
système réalisé sur la base d'éléments E/E/PE, exécutant des fonctions d'I&C de la centrale ainsi que des fonctions de service et de surveillance liées au fonctionnement du système lui-même

Note 1 à l'article: Le terme est utilisé comme terme général comprenant tous les éléments du système, tels que les alimentations électriques, les capteurs et autres dispositifs d'entrée, les bus de données et autres chemins de communication, les interfaces vers les actionneurs et autres dispositifs de sortie. Les différentes fonctions d'un système peuvent utiliser des ressources dédiées ou partagées.

Note 2 à l'article: Les éléments contenus dans un système d'I&C donné sont définis dans la spécification des limites de ce système.

Note 3 à l'article: Voir aussi la définition d'élément E/E/PE et les notes associées.

Note 4 à l'article: Selon leurs fonctionnalités propres, l'AIEA fait la distinction entre les systèmes de contrôle et de commande, les systèmes d'IHM, les systèmes de verrouillage et les systèmes de protection.

3.27
intégration
agrégation et vérification progressives des composant pour former un système complet

3.28
bibliothèque
ensemble d'éléments logiciels connexes contenus dans un fichier unique mais sélectionnés individuellement pour inclusion dans le produit logiciel final

[SOURCE: IEC 60880:2006, 3.24]

3.29
mode de fonctionnement
état de fonctionnement d'un élément qui dans ce cas adopte un comportement opérationnel particulier

EXEMPLE: Les modes d'initialisation, normal ou dégradé adopté en cas d'erreur dans l'élément.

3.30
logiciel système opérationnel
logiciel s'exécutant sur le processeur cible pendant le fonctionnement du système

EXEMPLE: Système d'exploitation, gestionnaires d'entrées/sorties et de communication, gestion des exceptions, bibliothèques d'application logicielles, auto-surveillance, gestion des redondances et de la dégradation progressive.

3.31
paramètre
donnée gouvernant le comportement du système d'I&C et / ou de son logiciel, et pouvant être modifiée par les opérateurs durant l'exploitation

3.32
logiciel prédéveloppé
logiciel qui existe déjà, est disponible en tant que produit commercial ou propriétaire, et dont l'utilisation est envisagée

Note 1 à l'article: Dans le présent document, les logiciels prédéveloppés sont de deux types:

a) le logiciel système opérationnel complet,

b) les composants logiciels.

Note 2 à l'article: Les logiciels prédéveloppés peuvent être répartis entre les logiciels qui n'ont pas été spécifiquement développés pour un environnement matériel particulier et les logiciels intégrés dans des composants matériels et qui sont à utiliser en association avec ces matériels.

Note 3 à l'article: Dans le présent document, ce terme ne couvre pas les outils logiciels, même si ceux-ci sont prédéveloppés.

Note 4 à l'article: Le logiciel d'application est spécifique à l'installation, ainsi il ne peut pas être considéré comme du logiciel prédéveloppé.

[SOURCE: IEC 60880:2006, 3.28, modifié (notes à l'article ajoutées)]

3.33
constituant prédéveloppé
constituant matériel ou logiciel ou programmé qui existe déjà, qui est disponible comme produit commercial ou propriétaire, et dont l'utilisation est envisagée

Note 1 à l'article: Cette définition est fournie par souci de cohérence avec les termes et définitions de l'IEC 61513:2011 mais n'est pas utilisée. Dans le présent document dédié au logiciel, le terme logiciel prédéveloppé est utilisé.

[SOURCE: IEC 61513:2011, 3.36, modifié (note à l'article modifiée)]

3.34
élément numérique programmable
élément qui s'appuie sur des instructions logicielles ou une logique programmable pour accomplir une fonction

Note 1 à l'article: Le terme «élément» peut être remplacé par le terme «système», «équipement» ou «dispositif».

Note 2 à l'article: Les principaux éléments numériques programmables sont les éléments informatisés et les éléments logiques programmables.

Note 3 à l'article: Ce terme utilisé par l'IEC SC 45A équivaut au terme «élément électronique programmable» utilisé dans l'IEC 61508.

3.35
élément à logique programmable
élément qui s'appuie sur circuit intégré composé d'éléments logiques avec un motif d'interconnexions, dont des parties sont programmables par l'utilisateur

Note 1 à l'article: Le terme «élément» peut être remplacé par les termes «système», «équipement» ou «dispositif».

Note 2 à l'article: Un élément à logique programmable est une sorte d'élément numérique programmable.

Note 3 à l'article: Voir aussi la définition d'élément E/E/PE et les notes associées.

3.36
auto-surveillance
test automatique des performances matérielles et de la cohérence logicielle d'un système d'I&C informatisé

[SOURCE: IEC 60671:2007, 3.8]

3.37
logiciel
programmes (ensembles ordonnés d'instructions), données, règles et toute documentation associée relatifs au fonctionnement d'un système informatisé

[SOURCE: IEC 61513:2011, 3.51]

3.38
composant logiciel
une des entités constituent un logiciel complet. Les composants logiciels doivent être intégrés pour former le logiciel complet

Note 1 à l'article: Dans le présent document, un élément logiciel prédéveloppé peut être considéré comme un composant logiciel seulement s'il est intégré dans un logiciel plus important pour former un logiciel opérationnel complet. En particulier, les vérifications et validation du logiciel opérationnel complet sont à réaliser avec les composants logiciels embarqués. L'intégration peut être faite dans un logiciel qui s'exécute sur un processeur unique, pour par exemple les systèmes d'exploitation temps réels ou bibliothèques. L'intégration peut être aussi faite dans un logiciel qui s'exécute en une coopération étroite répartie sur plusieurs processeurs, par exemple le microprogramme des modules de communication ou des modules d'entrée/sortie.

3.39
développement du logiciel
toutes les activités du cycle de vie du logiciel conduisant à la création du logiciel d'un système d'I&C ou d'un produit logiciel et qui couvrent toutes les phases depuis la spécification d'exigences jusqu'à la validation et l'installation sur le site

3.40
modification du logiciel
changement dans un document ou des documents déjà approuvés conduisant à un changement dans le code exécutable

Note 1 à l'article: Une modification du logiciel peut être effectuée durant le développement initial (par exemple pour éliminer des défauts mis en évidence dans les phases finales du développement) ou après la mise en service du logiciel.

[SOURCE: IEC 60880:2006, 3.36]

3.41
cycle de vie et de sûreté du logiciel
activités nécessaires au développement et à l'exploitation du logiciel d'un système d'I&C important pour la sûreté. Elles couvrent la période allant de la spécification des exigences sur le logiciel jusqu'au au retrait de service du logiciel

[SOURCE: IEC 60880:2006, 3.37]

3.42
validation du logiciel
test et évaluation d'un logiciel intégré pour s'assurer de sa conformité aux spécifications de fonctionnalité, de performance et d'interface imposées par les exigences sur le système d'I&C

Note 1 à l'article: Dans le présent document la validation du logiciel est considérée comme une partie de la validation du système.

3.43
analyse statique
processus d'évaluation d'un système ou d'un composant basé sur sa forme, sa structure, son contenu ou sa documentation. S'oppose à l'analyse dynamique

[SOURCE: IEC 60880:2006, 3.40]

3.44
logiciel système
logiciel conçu pour un système programmé particulier ou pour une famille de systèmes programmés afin de faciliter le fonctionnement et la maintenance de ce système et des programmes connexes, par exemple systèmes d'exploitation, ordinateurs, utilitaires. Le logiciel système est généralement composé de logiciels systèmes opérationnels et de logiciels de soutien

Note 1 à l'article: Logiciels systèmes opérationnels: logiciels fonctionnant sur le processeur cible pendant le fonctionnement du système. Par exemple: système d'exploitation, gestionnaires d'entrée/sortie et de communication, gestion des exceptions, bibliothèques d'application logicielles, auto-surveillance, gestion de la redondance et de la dégradation progressive.

Note 2 à l'article: Logiciels de soutien: logiciels d'aide au développement, aux essais ou à la maintenance des autres logiciels et du système tels que les compilateurs, les générateurs de codes, l'éditeur graphique, le diagnostic hors-ligne, les outils de vérification et de validation, etc.

Note 3 à l'article: Voir également «logiciel d'application».

[SOURCE: IEC 61513:2011, 3.58, modifié (notes 2,3 et 4 à l'article ajoutées)]

3.45
validation système
confirmation par examen et apport d'autres éléments justificatifs qu'un système satisfait à la totalité des exigences spécifiées (fonctionnalités, temps de réponse, tolérance aux fautes, robustesse)

Note 1 à l'article: L'édition 2016 du Glossaire de Sûreté de l'AIEA donne les deux définitions suivantes:

Validation: Processus visant à déterminer si un produit ou un service est capable de remplir sa fonction prévue de façon satisfaisante. La validation peut faire intervenir plus d'élément de jugement que la vérification.

Validation du système informatisé: Processus consistant à tester et évaluer le système informatisé intégré (matériel et logiciel) afin de garantir sa conformité par rapport aux exigences fonctionnelles, aux exigences relatives aux performances et à celles concernant les interfaces.

Le premier point qui doit être relevé pour ce qui concerne la définition de validation système est que celle-ci est un cas spécifique de validation, qu'elle fait référence à un produit particulier, à savoir la validation d'un système d'I&C. Ceci est cohérent avec la définition de l'AIEA. Deuxièmement, la définition IEC spécifie la référence de validation, à savoir les spécifications d'exigences alors que la définition de l'AIEA fait seulement référence à la fonction attendue.

[SOURCE: IEC 61513:2011, 3.59]

3.46
défaut systématique
défaut relié de façon déterministe à une certaine cause, ne pouvant être éliminée que par une modification de la conception ou du processus de fabrication, des procédures d'exploitation, de la documentation ou d'autres facteurs appropriés

[SOURCE: IEC 61513:2011, 3.60]

3.47
vérification
confirmation par examen et apport d'éléments objectifs que les résultats d'une activité sont conformes aux objectifs et exigences établis pour cette activité

[SOURCE: IEC 61513:2011, 3.62]

4 Symboles et termes abrégés

CB	Informatisé (Computer-Based)
DCC	Défaillance de Cause Commune
EPROM	Erasable Programmable Read Only Memory
IHM	Interface homme machine
HDL	Langage de description de matériel (Hardware Description Language)
HPD	Circuit intégré programmé en HDL (HDL-Programmed Device)
I&C	Instrumentation et contrôle-commande
CNP	Centrale nucléaire de puissance

5 Concepts et présupposés

5.1 Généralité

L'Article 5 présente les principaux concepts et présupposés relatifs à la nature et au développement du logiciel des systèmes d'I&C des classes de sûreté 2 et 3, et sur lesquels le texte normatif repose.

5.2 Types de logiciels

La Figure 1 illustre la gamme des services offerts par les logiciels d'un système d'I&C ou d'une architecture d'I&C typiques. Les logiciels sont souvent répartis entre logiciel système et logiciel d'application. Le logiciel système est lui-même divisé en logiciel système opérationnel, qui est embarqué dans les systèmes d'I&C classés de sûreté, et en logiciel de soutien (ou outils logiciels) qui est hors-ligne ou embarqué dans des systèmes de service non classés de sûreté. Du logiciel peut aussi être trouvé dans des équipements spécialisés tels que des capteurs et des actionneurs, des équipements de communication et des onduleurs.

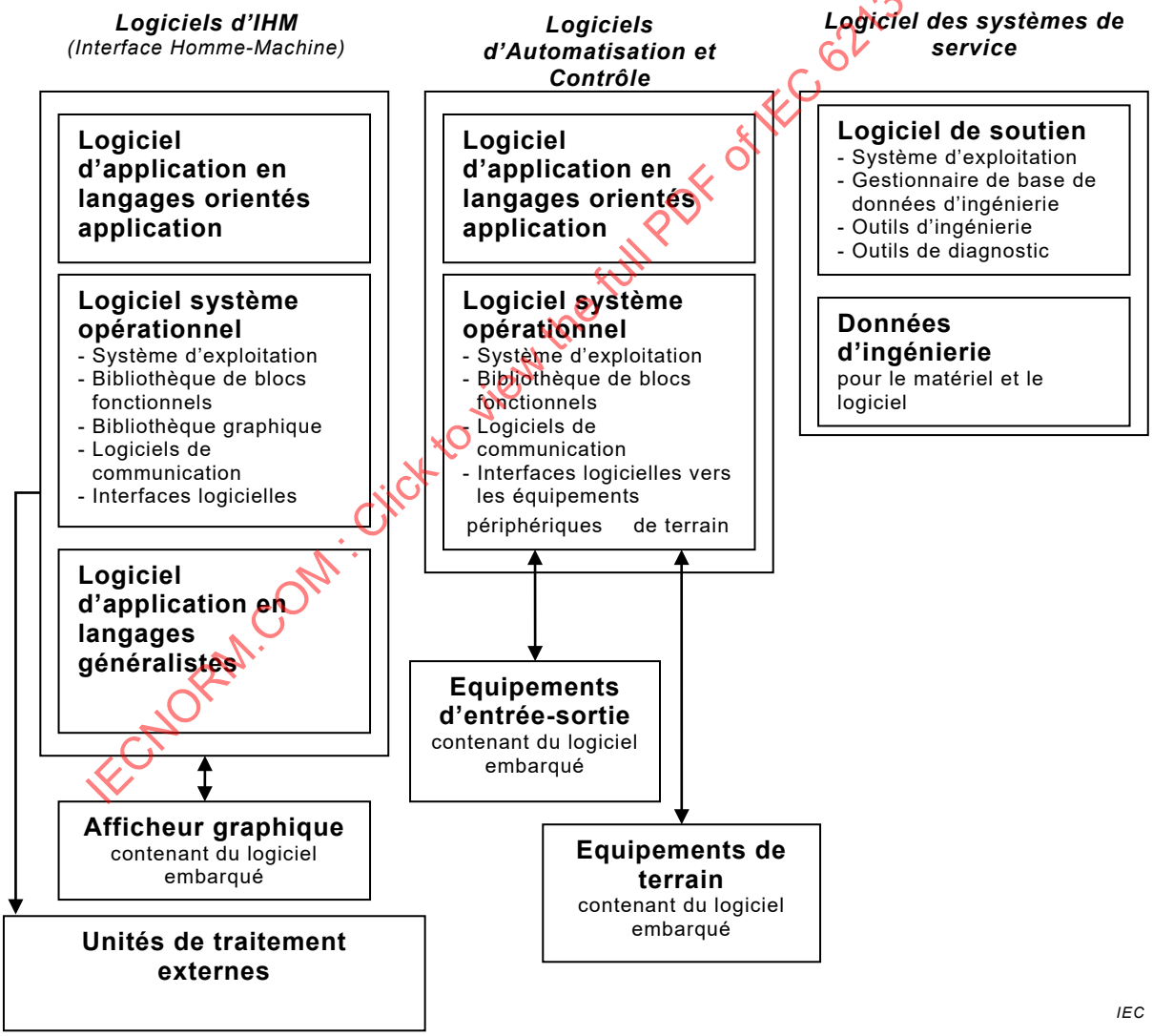


Figure 1 – Composants logiciels typiques d'un système d'I&C informatisé

Le logiciel d'un système d'I&C peut aussi être divisé en logiciel prédéveloppé (offrant le plus souvent des fonctions utiles pour une variété de systèmes d'I&C) et en logiciel nouveau (développé le plus souvent pour les besoins spécifiques d'un système d'I&C). Les exigences du présent document qui couvrent les questions pertinentes concernant les logiciels

nouveaux peuvent également être rétrospectivement appliquées aux logiciels prédéveloppés. Dans certains cas, cependant, le présent document énonce des exigences de substitution pour traiter spécifiquement des questions pertinentes pour les logiciels prédéveloppés.

De nombreuses familles d'équipements incluent une large panoplie d'outils de développement orientés application permettant aux ingénieurs concevant la centrale ou les systèmes élémentaires de spécifier leurs exigences graphiquement. Des outils peuvent alors traduire automatiquement les graphiques représentant des programmes en logiciel d'application exécutable. Lorsque ces outils sont d'une qualité appropriée, il est admis que cette approche permet de réduire les risques de défaut.

5.3 Types de données de configuration

La conception de nombreux systèmes fait largement appel à des données de configuration. Une donnée de configuration peut être liée au logiciel système opérationnel ou au logiciel d'application. Les données de configuration liées au logiciel d'application sont principalement des données d'ingénierie résultant de la conception de la centrale et sont souvent produites pour l'essentiel par des concepteurs de centrale qui n'ont pas besoin d'une expérience particulière en génie logiciel. Les données de configuration peuvent être divisées en:

- données qui ne peuvent être modifiées en ligne par les opérateurs de la centrale, et qui sont soumises aux mêmes exigences que le reste du logiciel;
- paramètres qui peuvent être modifiés par les opérateurs durant l'exploitation de la centrale (par exemple les seuils d'alarme, les points de consigne, les données d'étalonnage pour calibrer l'instrumentation) et qui font l'objet d'exigences particulières.

5.4 Cycles de vie et de sûreté du logiciel et du système

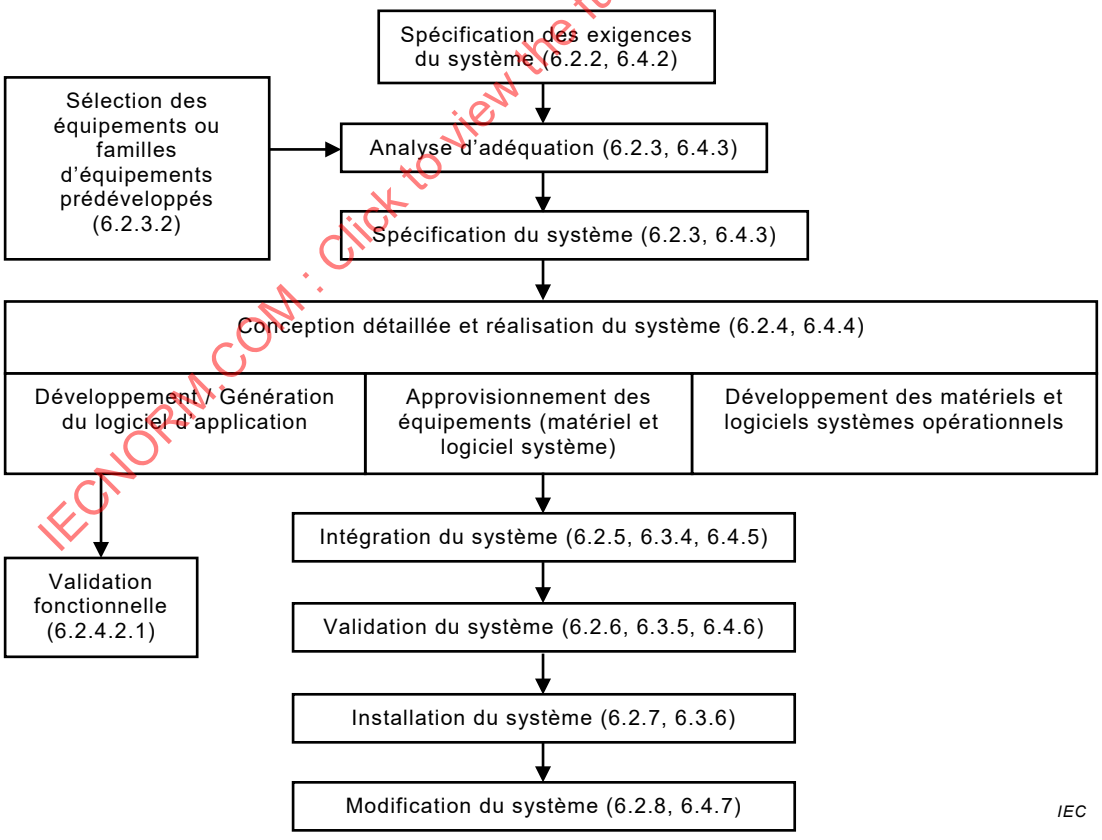


Figure 2 – Activités du cycle de vie de sûreté du système (selon l'IEC 61513:2011)

Le logiciel contribue en général fortement aux fonctions réalisées par le système d'I&C. Il peut aussi contribuer à des fonctions ajoutées car nécessaire au fonctionnement du système lui-même (initialisation et surveillance du matériel, communication entre sous-systèmes et