Publication 61508-3 de la CEI
(Première édition – 1998)

Sécurité fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité –

Partie 3: Prescriptions concernant les logiciels

IEC Publication 61508-3
(First edition – 1998)

Functional safety of electrical/electronic/programmable electronic safety-related systems –

Part 3: Software requirements

CORRIGENDUM

Page 14

Remplacer le premier alinéa de l'article 1.2 existant par le texte amendé suivant:

1.2 Les parties 1, 2, 3 et 4 de la présente norme sont des publications fondamentales de sécurité, bien qu'un tel statut ne soit pas applicable dans le contexte des systèmes E/E/PE de faible complexité relatifs à la sécurité (voir 3.4.4 de la partie 4). En tant que publications fondamentales de sécurité, ces normes sont prévues pour être utilisées par les comités techniques pour la préparation des normes selon les principes contenus dans le *Guide CEI 104* et le *Guide ISO/CEI 51*. Les parties 1, 2, 3 et 4 sont également destinées à être utilisées comme publications autonomes.

Une des responsabilités incombant à un comité technique est, dans la mesure du possible, d'utiliser les publications fondamentales de sécurité pour la préparation de ses publications. Dans ce contexte les prescriptions, les méthodes d'essai ou conditions d'essai de cette publication fondamentale de sécurité ne s'appliquent que si elles sont indiquées spécifiquement ou incluses dans les publications préparées par ces comités techniques.

Page 15

Replace the existing first paragraph of clause 1.2 by the following amended text:

1.2 Parts 1, 2, 3 and 4 of this standard are basic safety publications, although this status does not apply in the context of low complexity E/E/PE safety-related systems (see 3.4.4 of part 4). As basic safety publications, they are intended for use by technical committees in the preparation of standards in accordance with the principles contained in *IEC Guide 104* and *ISO/IEC Guide 51*. Parts 1, 2, 3, and 4 are also intended for use as stand-alone publications.

One of the responsibilities of a technical committee is, wherever applicable, to make use of basic safety publications in the preparation of its publications. In this context, the requirements, test methods or test conditions of this basic safety publication will not apply unless specifically referred to or included in the publications prepared by those technical committees.

NORME INTERNATIONALE

INTERNATIONAL STANDARD

CEI
IEC
61508-3

Première édition
First edition
1998-12

Sécurité fonctionnelle des systèmes électriques/ électroniques/électroniques programmables relatifs à la sécurité –

Partie 3: Prescriptions concernant les logiciels

Functional safety of electrical/electronic/programmable electronic safety-related systems –

Part 3: Software requirements

Numéro de référence
Reference number
CEI/IEC 61508-3:1998

Numéros des publications

Depuis le 1er janvier 1997, les publications de la CEI sont numérotées à partir de 60000.

Publications consolidées

Les versions consolidées de certaines publications de la CEI incorporant les amendements sont disponibles. Par exemple, les numéros d'édition 1.0, 1.1 et 1.2 indiquent respectivement la publication de base, la publication de base incorporant l'amendement 1, et la publication de base incorporant les amendements 1 et 2.

Validité de la présente publication

Le contenu technique des publications de la CEI est constamment revu par la CEI afin qu'il reflète l'état actuel de la technique.

Des renseignements relatifs à la date de reconfirmation de la publication sont disponibles dans le Catalogue de la CEI.

Les renseignements relatifs à des questions à l'étude et des travaux en cours entrepris par le comité technique qui a établi cette publication, ainsi que la liste des publications établies, se trouvent dans les documents ci-dessous:

- **«Site web» de la CEI***
- **Catalogue des publications de la CEI**
  Publié annuellement et mis à jour régulièrement (Catalogue en ligne)*
- **Bulletin de la CEI**
  Disponible à la fois au «site web» de la CEI* et comme périodique imprimé

Terminologie, symboles graphiques et littéraux

En ce qui concerne la terminologie générale, le lecteur se reportera à la CEI 60050: *Vocabulaire Electrotechnique International* (VEI).

Pour les symboles graphiques, les symboles littéraux et les signes d'usage général approuvés par la CEI, le lecteur consultera la CEI 60027: *Symboles littéraux à utiliser en électrotechnique,* la CEI 60417: *Symboles graphiques utilisables sur le matériel. Index, relevé et compilation des feuilles individuelles,* et la CEI 60617: *Symboles graphiques pour schémas.*

* Voir adresse «site web» sur la page de titre.

Numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series.

Consolidated publications

Consolidated versions of some IEC publications including amendments are available. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Validity of this publication

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology.

Information relating to the date of the reconfirmation of the publication is available in the IEC catalogue.

Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is to be found at the following IEC sources:

- **IEC web site***
- **Catalogue of IEC publications**
  Published yearly with regular updates (On-line catalogue)*
- **IEC Bulletin**
  Available both at the IEC web site* and as a printed periodical

Terminology, graphical and letter symbols

For general terminology, readers are referred to IEC 60050: *International Electrotechnical Vocabulary* (IEV).

For graphical symbols, and letter symbols and signs approved by the IEC for general use, readers are referred to publications IEC 60027: *Letter symbols to be used in electrical technology*, IEC 60417: *Graphical symbols for use on equipment. Index, survey and compilation of the single sheets* and IEC 60617: *Graphical symbols for diagrams.*

* See web site address on title page.

NORME INTERNATIONALE

INTERNATIONAL STANDARD

CEI
IEC
61508-3

Première édition
First edition
1998-12

Sécurité fonctionnelle des systèmes électriques/ électroniques/électroniques programmables relatifs à la sécurité –

Partie 3: Prescriptions concernant les logiciels

Functional safety of electrical/electronic/programmable electronic safety-related systems –

Part 3: Software requirements

International Electrotechnical Commission 3, rue de Varembé Geneva, Switzerland
Telefax: +41 22 919 0300 e-mail: inmail@iec.ch IEC web site http://www.iec.ch

Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

CODE PRIX
PRICE CODE X

Pour prix, voir catalogue en vigueur
For price, see current catalogue

SOMMAIRE

CONTENTS

Tableaux Pages

Figures

Table Page

Figures

SÉCURITÉ FONCTIONNELLE DES SYSTÈMES ÉLECTRIQUES/ÉLECTRONIQUES/ÉLECTRONIQUES PROGRAMMABLES RELATIFS À LA SÉCURITÉ –

Partie 3: Prescriptions concernant les logiciels

AVANT-PROPOS

1) La CEI (Commission Electrotechnique Internationale) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI 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, la CEI, entre autres activités, publie des Normes internationales. 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 la CEI, participent également aux travaux. La CEI 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 la CEI concernant les questions techniques représentent, dans la mesure du possible un accord international sur les sujets étudiés, étant donné que les Comités nationaux intéressés sont représentés dans chaque comité d'études.
3) Les documents produits se présentent sous la forme de recommandations internationales. Ils sont publiés comme normes, rapports techniques ou guides et agréés comme tels par les Comités nationaux.
4) Dans le but d'encourager l'unification internationale, les Comités nationaux de la CEI s'engagent à appliquer de façon transparente, dans toute la mesure possible, les Normes internationales de la CEI dans leurs normes nationales et régionales. Toute divergence entre la norme de la CEI et la norme nationale ou régionale correspondante doit être indiquée en termes clairs dans cette dernière.
5) La CEI n'a fixé aucune procédure concernant le marquage comme indication d'approbation et sa responsabilité n'est pas engagée quand un matériel est déclaré conforme à l'une de ses normes.
6) L'attention est attirée sur le fait que certains des éléments de la présente Norme internationale peuvent faire l'objet de droits de propriété intellectuelle ou de droits analogues. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de propriété et de ne pas avoir signalé leur existence.

La Norme internationale CEI 61508-3 a été établie par le sous-comité 65A: Aspects systèmes, du comité d'études 65 de la CEI: Mesure et commande dans les processus industriels.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65A/269/FDIS	65A/277/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.

Les annexes A et B font partie intégrante de cette norme.
L'annexe C est donnée uniquement à titre d'information.

La CEI 61508 est composée des parties suivantes, regroupées sous le titre général *Sécurité fonctionnelle des systèmes de sécurité électriques/électroniques/électroniques programmables*:

- Partie 1: Prescriptions générales
- Partie 2: Prescriptions pour les systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité
- Partie 3: Prescriptions concernant les logiciels
- Partie 4: Définitions et abréviations
- Partie 5: Exemples de méthodes pour la détermination des niveaux d'intégrité de sécurité
- Partie 6: Lignes directrices pour l'application de la CEI 61508-2 et de la CEI 61508-3
- Partie 7: Présentation de techniques et mesures

FUNCTIONAL SAFETY OF ELECTRICAL/ELECTRONIC/PROGRAMMABLE ELECTRONIC SAFETY-RELATED SYSTEMS –

Part 3: Software requirements

FOREWORD

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the 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, the IEC publishes International Standards. 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. The 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 the 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 National Committees.
3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical reports or guides and they are accepted by the National Committees in that sense.
4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
6) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61508-3 has been prepared by subcommittee 65A: System aspects, of IEC technical committee 65: Industrial-process measurement and control.

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

FDIS	Report on voting
65A/269/FDIS	65A/277/RVD

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

Annexes A and B form an integral part of this standard.
Annex C is for information only.

IEC 61508 consists of the following parts, under the general title *Functional safety of electrical/electronic/programmable electronic safety-related systems*:

- Part 1: General requirements
- Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems
- Part 3: Software requirements
- Part 4: Definitions and abbreviations
- Part 5: Examples of methods for the determination of safety integrity levels
- Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3
- Part 7: Overview of techniques and measures

INTRODUCTION

Les systèmes constitués de composants électriques et/ou électroniques sont utilisés depuis des années pour exécuter des fonctions liées à la sécurité dans la plupart des secteurs d'application. Des systèmes à base informatique (généralement désignés par l'appellation: «Systèmes électroniques programmables (PES)») sont utilisés dans tous les secteurs d'application pour exécuter des fonctions non liées à la sécurité, mais aussi de plus en plus souvent liées à la sécurité. Si l'on veut exploiter efficacement, et en toute sécurité, la technologie des systèmes informatiques, il est indispensable de fournir à tous les responsables suffisamment d'éléments liés à la sécurité pour les guider dans leurs prises de décisions.

La présente Norme internationale présente une approche générique de toutes les activités liées au cycle de vie de sécurité concernant les systèmes constitués de composants électriques et/ou électroniques et/ou électroniques programmables (E/E/PES) destinés à exécuter des fonctions de sécurité. Cette approche unifiée a été adoptée afin de développer une politique technique rationnelle et logique concernant tous les appareils électriques liés à la sécurité. L'un des principaux objectifs poursuivis consiste à faciliter l'élaboration de normes destinées à chaque secteur d'application.

Dans la plupart des cas, la sécurité est obtenue par un certain nombre de systèmes de protection fondés sur diverses technologies (par exemple mécanique, hydraulique, pneumatique, électrique, électronique, électronique programmable). En conséquence, il faut que toute stratégie de sécurité prenne non seulement en compte tous les éléments d'un système individuel (par exemple les capteurs, les appareils de commande, les actionneurs), mais aussi qu'elle considère tous les systèmes de sécurité comme des éléments individuels d'un ensemble complexe. C'est pourquoi cette Norme internationale, bien que traitant essentiellement des E/E/PES, fournit néanmoins un cadre de sécurité susceptible de concerner les systèmes de sécurité basés sur des technologies différentes.

Personne n'ignore la grande variété des applications E/E/PES. Celles-ci recouvrent, à des degrés de complexité très divers, un fort potentiel de danger et de risques dans tous les secteurs d'application. Pour chaque application, la nature exacte des mesures de sécurité envisagées dépendra de plusieurs facteurs propres à l'application. La présente Norme internationale, de par son caractère général, rendra désormais possible la prescription de ces mesures dans des normes internationales spécifiques à chaque secteur d'application.

La présente Norme internationale

- concerne toutes les phases appropriées du cycle de vie de sécurité des E/E/PES et de leurs logiciels (depuis la conceptualisation initiale jusqu'au déclassement, en passant par la création, l'installation, la mise en service et l'entretien) lorsque les E/E/PES exécutent des fonctions de sécurité;
- a été conçue dans le souci de l'évolution rapide des technologies; le cadre est suffisamment solide et étendu pour pourvoir aux évolutions futures;
- permet l'élaboration de normes internationales par secteur d'application concernant les E/E/PES relatifs à la sécurité. L'élaboration de normes internationales par secteur d'application à partir de la présente Norme internationale devrait permettre d'atteindre un haut niveau de cohérence (par exemple pour ce qui est des principes sous-jacents, de la terminologie, etc.) à la fois au sein de chaque secteur d'application, et d'un secteur à l'autre. La conséquence en est une amélioration en termes de sécurité et de bénéfices économiques;
- fournit une méthode de développement des prescriptions de sécurité nécessaires pour réaliser la sécurité fonctionnelle des systèmes de sécurité E/E/PE;
- utilise des niveaux d'intégrité de sécurité afin de spécifier les niveaux objectifs d'intégrité de sécurité des fonctions de sécurité à réaliser par les systèmes de sécurité E/E/PE;

INTRODUCTION

Systems comprised of electrical and/or electronic components have been used for many years to perform safety functions in most application sectors. Computer-based systems (generically referred to as programmable electronic systems (PESs)) are being used in all application sectors to perform non-safety functions and, increasingly, to perform safety functions. If computer system technology is to be effectively and safely exploited, it is essential that those responsible for making decisions have sufficient guidance on the safety aspects on which to make those decisions.

This International Standard sets out a generic approach for all safety lifecycle activities for systems comprised of electrical and/or electronic and/or programmable electronic components (electrical/electronic/ programmable electronic systems (E/E/PESs)) that are used to perform safety functions. This unified approach has been adopted in order that a rational and consistent technical policy be developed for all electrically-based safety-related systems. A major objective is to facilitate the development of application sector standards.

In most situations, safety is achieved by a number of protective systems which rely on many technologies (for example mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic). Any safety strategy must therefore consider not only all the elements within an individual system (for example sensors, controlling devices and actuators), but also all the safety-related systems making up the total combination of safety-related systems. Therefore, while this International Standard is concerned with electrical/electronic/ programmable electronic (E/E/PE) safety-related systems, it may also provide a framework within which safety-related systems based on other technologies may be considered.

It is recognized that there is a great variety of E/E/PES applications in a variety of application sectors and covering a wide range of complexity, hazard and risk potentials. In any particular application, the required safety measures will be dependent on many factors specific to the application. This International Standard, by being generic, will enable such measures to be formulated in future application sector international standards.

This International Standard

- considers all relevant overall, E/E/PES and software safety lifecycle phases (for example, from initial concept, through design, implementation, operation and maintenance to decommissioning) when E/E/PESs are used to perform safety functions;
- has been conceived with a rapidly developing technology in mind; the framework is sufficiently robust and comprehensive to cater for future developments;
- enables application sector international standards, dealing with safety-related E/E/PESs, to be developed; the development of application sector international standards, within the framework of this International Standard, should lead to a high level of consistency (for example, of underlying principles, terminology etc.) both within application sectors and across application sectors; this will have both safety and economic benefits;
- provides a method for the development of the safety requirements specification necessary to achieve the required functional safety for E/E/PE safety-related systems;
- uses safety integrity levels for specifying the target level of safety integrity for the safety functions to be implemented by the E/E/PE safety-related systems;

- adopte une approche basée sur le risque encouru pour déterminer les prescriptions de niveaux d'intégrité de sécurité;
- fixe des objectifs numériques pour les mesures de défaillances des systèmes de sécurité E/E/PE qui sont en rapport avec les niveaux d'intégrité de sécurité;
- fixe une limite inférieure pour les mesures de défaillances, dans le cas d'un mode de défaillance dangereux, cette limite pouvant être exigée pour un système de sécurité E/E/PE unique. Dans le cas d'un système de sécurité E/E/PE fonctionnant
  - dans un mode de fonctionnement faible demande, la limite inférieure est fixée à une probabilité de défaillance de 10^{-5} par heure afin que les fonctions pour lesquelles le système a été conçu soient exécutées lorsqu'elles sont requises,
  - dans un mode de fonctionnement forte demande, ou continu, la limite inférieure est fixée à une probabilité de défaillance dangereuse de 10^{-9} par heure;

  NOTE – Un système E/E/PE de sécurité unique ne signifie pas nécessairement architecture à une seule voie.
- adopte une large gamme de principes, techniques et mesures pour la réalisation de la sécurité fonctionnelle des systèmes de sécurité E/E/PE, mais n'utilise pas le concept de sécurité intrinsèque qui a un sens particulier lorsque les modes de défaillances sont bien définis et que le niveau de complexité est relativement faible. Ce concept a été considéré comme non approprié en raison de l'immense gamme de complexité des systèmes de sécurité E/E/PE qui entrent dans le domaine d'application de la présente norme.

- adopts a risk-based approach for the determination of the safety integrity level requirements;
- sets numerical target failure measures for E/E/PE safety-related systems which are linked to the safety integrity levels;
- sets a lower limit on the target failure measures, in a dangerous mode of failure, that can be claimed for a single E/E/PE safety-related system; for E/E/PE safety-related systems operating in
  - a low demand mode of operation, the lower limit is set at an average probability of failure of 10^{-5} to perform its design function on demand,
  - a high demand or continuous mode of operation, the lower limit is set at a probability of a dangerous failure of 10^{-9} per hour;

  NOTE – A single E/E/PE safety-related system does not necessarily mean a single-channel architecture.
- adopts a broad range of principles, techniques and measures to achieve functional safety for E/E/PE safety-related systems, but does not use the concept of fail safe, which may be of value when the failure modes are well defined and the level of complexity is relatively low. The concept of fail safe was considered inappropriate because of the full range of complexity of E/E/PE safety-related systems that are within the scope of the standard.

SÉCURITÉ FONCTIONNELLE DES SYSTÈMES ÉLECTRIQUES/ÉLECTRONIQUES/ÉLECTRONIQUES PROGRAMMABLES RELATIFS À LA SÉCURITÉ –

Partie 3: Prescriptions concernant les logiciels

1 Domaine d'application

1.1 La présente partie de la CEI 61508

a) est prévue pour n'être utilisée qu'après s'être assuré d'une compréhension parfaite de la CEI 61508-1 et de la CEI 61508-2;

b) s'applique à tout logiciel faisant partie d'un système relatif à la sécurité, ou utilisé pour développer un système relatif à la sécurité entrant dans le domaine de la CEI 61508-1 et de la CEI 61508-2. Ce type de logiciel est désigné par le terme «logiciel relatif à la sécurité»;

- Les logiciels relatifs à la sécurité comprennent les systèmes d'exploitation, les logiciels système, les logiciels des réseaux de communication, les fonctions d'interface homme-machine, les outils supports et les micrologiciels («firmware»), ainsi que la programmation d'applications.
- Les programmes d'applications comprennent les programmes de haut niveau, de bas niveau et les programmes spécifiques dans des langages de variabilité limitée (voir 3.2.7 de la CEI 61508-4).

c) nécessite que les fonctions de sécurité du logiciel et les niveaux d'intégrité de sécurité du logiciel soient précisés.

NOTE 1 – Si cela a déjà été réalisé dans le cadre de la spécification des systèmes E/E/PE relatifs à la sécurité (voir 7.2 de la CEI 61508-2), il n'est pas nécessaire de le répéter dans la présente partie.

NOTE 2 – Spécifier les fonctions de sécurité du logiciel et les niveaux d'intégrité de sécurité du logiciel est une procédure itérative; voir les figures 2 et 6.

NOTE 3 – Voir l'article 5 et l'annexe A de la CEI 61508-1 pour la structure de la documentation. Cette structure peut tenir compte des procédures internes de la société et des procédés de travail des secteurs d'application spécifiques.

d) établit des prescriptions concernant les phases et activités du cycle de vie de sécurité qui doivent être appliquées durant la conception et développement du logiciel relatif à la sécurité (modèle de cycle de vie de sécurité du logiciel). Ces prescriptions comprennent l'application de mesures et de techniques qui suivent une gradation basée sur le niveau d'intégrité de sécurité, afin d'éviter et de maîtriser les défauts et défaillances du logiciel;

e) fournit les prescriptions pour les informations relatives à la validation de la sécurité du logiciel et devant être transmises à l'organisation en charge de l'intégration E/E/PES;

f) fournit les prescriptions pour la préparation des informations et procédures concernant le logiciel requis par l'utilisateur pour le fonctionnement et la maintenance d'un système relatif à la sécurité;

g) fournit les prescriptions devant être observées par l'organisation en charge des modifications du logiciel relatif à la sécurité;

h) fournit, en accord avec la CEI 61508-1 et CEI 61508-2, les prescriptions pour les outils supports tels que les outils de conception et développement, les traducteurs de langage, les outils de test et de mise au point et les outils de gestion de configuration.

NOTE 4 – Les figures 4 et 6 montrent la relation entre la CEI 61508-2 et la CEI 61508-3.

FUNCTIONAL SAFETY OF ELECTRICAL/ELECTRONIC/PROGRAMMABLE ELECTRONIC SAFETY-RELATED SYSTEMS –

Part 3: Software requirements

1 Scope

1.1 This part of IEC 61508

a) is intended to be utilised only after a thorough understanding of IEC 61508-1 and IEC 61508-2;

b) applies to any software forming part of a safety-related system or used to develop a safety-related system within the scope of IEC 61508-1 and IEC 61508-2. Such software is termed safety-related software.

- Safety-related software includes operating systems, system software, software in communication networks, human-computer interface functions, support tools and firmware as well as application programs.
- Application programs include high level programs, low level programs and special purpose programs in limited variability languages (see 3.2.7 of IEC 61508-4).

c) requires that the software safety functions and software safety integrity levels are specified.

NOTE 1 – If this has already been done as part of the specification of the E/E/PE safety-related systems (see 7.2 of IEC 61508-2), then it does not have to be repeated in this part.

NOTE 2 – Specifying the software safety functions and software safety integrity levels is an iterative procedure; see figures 2 and 6.

NOTE 3 – See clause 5 and annex A of IEC 61508-1 for documentation structure. The documentation structure may take account of company procedures, and of the working practices of specific application sectors.

d) establishes requirements for safety lifecycle phases and activities which shall be applied during the design and development of the safety-related software (the software safety lifecycle model). These requirements include the application of measures and techniques, which are graded against the safety integrity level, for the avoidance of and control of faults and failures in the software.

e) provides requirements for information relating to the software safety validation to be passed to the organisation carrying out the E/E/PES integration.

f) provides requirements for the preparation of information and procedures concerning software needed by the user for the operation and maintenance of the E/E/PE safety-related system.

g) provides requirements to be met by the organisation carrying out modifications to safety-related software.

h) provides, in conjunction with IEC 61508-1 and IEC 61508-2, requirements for support tools such as development and design tools, language translators, testing and debugging tools, configuration management tools.

NOTE 4 – Figures 4 and 6 show the relationship between IEC 61508-2 and IEC 61508-3.

1.2 La CEI 61508-1, la CEI 61508-2, la CEI 61508-3 et la CEI 61508-4 sont des publications de sécurité de base, bien qu'un tel statut ne soit pas applicable dans le contexte des systèmes E/E/PE de faible complexité relatifs à la sécurité (voir 3.4.4 de la CEI 61508-4). En tant que publications de sécurité de base, ces normes sont prévues pour être utilisées par les comités techniques pour la préparation des normes selon les principes contenus dans le *Guide ISO/CEI 104* et le *Guide ISO/CEI 51*. Une des responsabilités incombant à un comité technique est, dans la mesure du possible, d'utiliser les publications de sécurité de base pour la préparation de ses propres publications. La CEI 61508 est également prévue pour être utilisée séparément.

NOTE – Aux Etats-Unis d'Amérique et au Canada, les normes nationales de sécurité des processus existantes, basées sur la CEI 61508 (par exemple l'ANSI/ISA S48.01-1996) peuvent être appliquées dans le domaine des processus, à la place de la CEI 61508, et cela jusqu'à ce que les normes internationales concernant la mise en œuvre de la CEI 61508 (soit la CEI 61511) dans le domaine des processus soient publiées.

1.3 La figure 1 montre la structure globale des parties 1 à 7 de la CEI 61508 et indique le rôle dévolu à la CEI 61508-3 pour assurer la sécurité fonctionnelle des systèmes E/E/PE relatifs à la sécurité. L'annexe A de la CEI 61508-6 décrit l'application de la CEI 61508-2 et de la CEI 61508-3.

1.2 IEC 61508-1, IEC 61508-2, IEC 61508-3 and IEC 61508-4 are basic safety publications, although this status does not apply in the context of low complexity E/E/PE safety-related systems (see 3.4.4 of IEC 61508-4). As basic safety publications, they are intended for use by technical committees in the preparation of standards in accordance with the principles contained in *IEC Guide 104* and *ISO/IEC Guide 51*. One of the responsibilities of a technical committee is, wherever applicable, to make use of basic safety publications in the preparation of its own publications. IEC 61508 is also intended for use as a stand-alone standard.

NOTE – In the USA and Canada, until the proposed process sector implementation of IEC 61508 (i.e. IEC 61511) is published as an international standard in the USA and Canada, existing national process safety standards based on IEC 61508 (i.e. ANSI/ISA S84.01-1996) can be applied to the process sector instead of IEC 61508.

1.3 Figure 1 shows the overall framework of parts 1 to 7 IEC 61508, and indicates the role that IEC 61508-3 plays in the achievement of functional safety for E/E/PE safety-related systems. Annex A of IEC 61508-6 describes the application of IEC 61508-2 and IEC 61508-3.

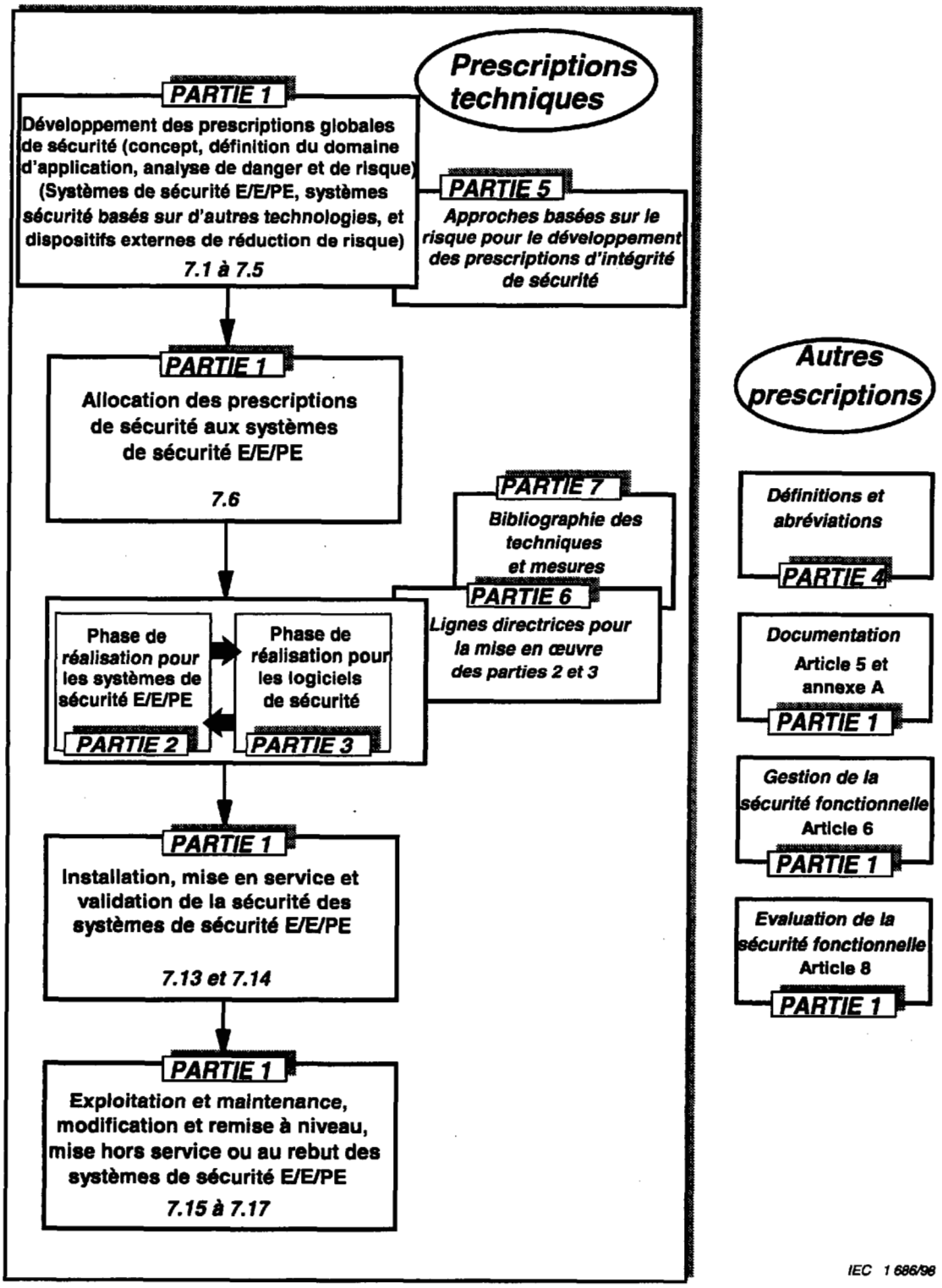


Figure 1 — Structure globale de la présente norme

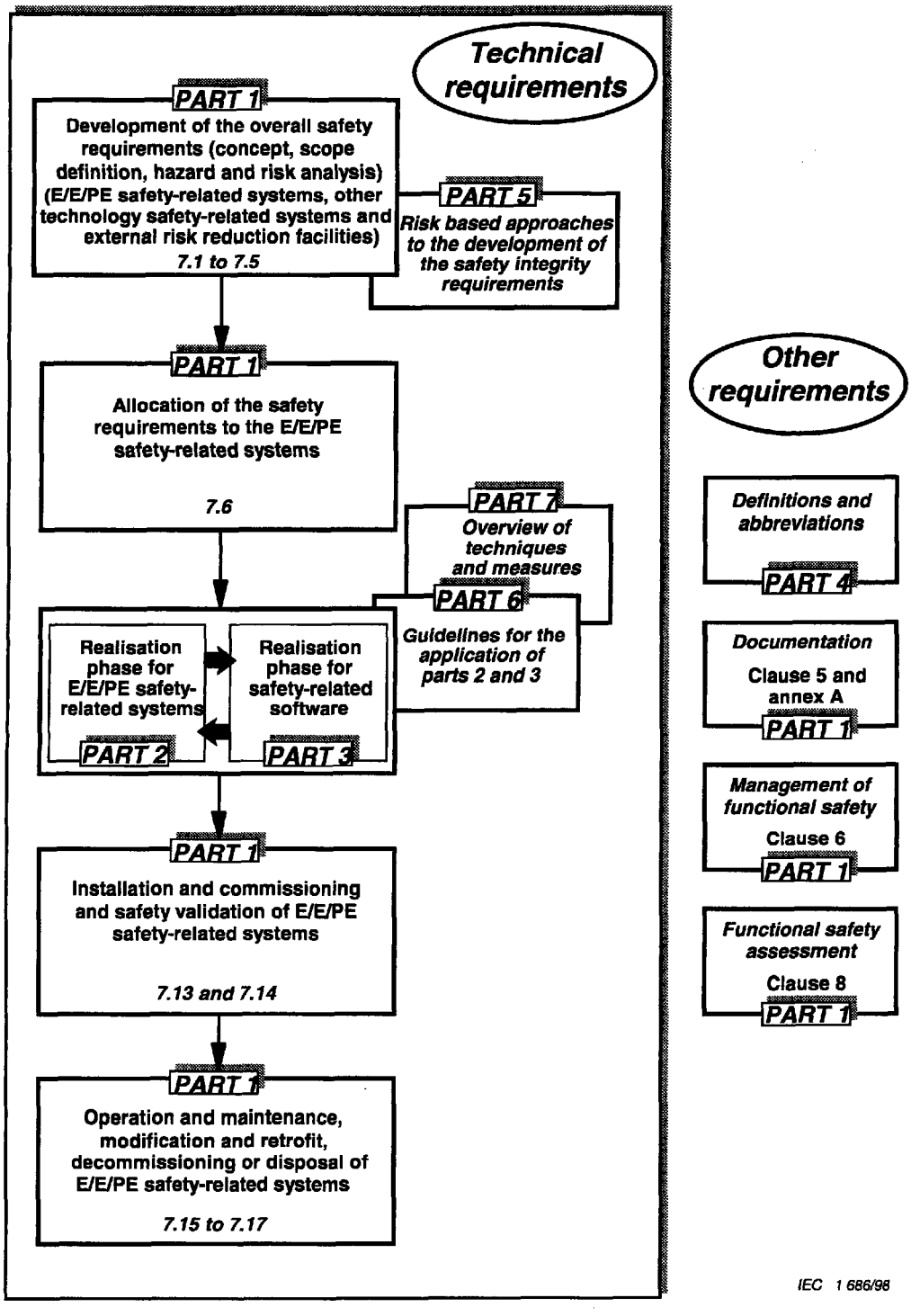


Figure 1 – Overall framework of this standard

2 Références normatives

Les documents normatifs suivants contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions valables pour la présente partie de la CEI 61508. Au moment de sa publication, les éditions indiquées étaient en vigueur. Tout document normatif est sujet à révision et les parties prenantes aux accords fondés sur la présente partie de la CEI 61508 sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des documents normatifs indiqués ci-après. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur.

CEI 61508-1:1998, *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 1: Prescriptions générales*

CEI 61508-2, — *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 2: Prescriptions pour les systèmes électriques/ électroniques/électroniques programmables relatifs à la sécurité* [1)]

CEI 61508-4:1998, *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 4: Définitions et abréviations*

CEI 61508-5:1998, *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 5: Exemples de méthodes pour la détermination des niveaux d'intégrité de sécurité*

CEI 61508-6, — *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 6: Lignes directrices pour l'application des parties 2 et 3* [1)]

CEI 61508-7, — *Sûreté fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 7: Présentation de techniques et mesures* [1)]

Guide ISO/CEI 51:1990, *Principes directeurs pour inclure dans les normes les aspects liés à la sécurité*

Guide CEI 104:1997, *Guide pour la rédaction des normes de sécurité et rôle des comités chargés de fonctions pilotes de sécurité et de fonctions groupées de sécurité*

3 Définitions et abréviations

Les définitions et abréviations utilisées dans la présente norme figurent dans la CEI 61508-4.

4 Conformité à la présente norme

Les prescriptions de conformité à la présente norme figurent à l'article 4 de la CEI 61508-1.

5 Documentation

Les objectifs et prescriptions concernant la documentation figurent à l'article 5 de la CEI 61508-1.

---

1) A publier.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of IEC 61508. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this part of IEC 61508 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

IEC 61508-1:1998, *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 1: General requirements*

IEC 61508-2, — *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 2: Requirements for electrical/electronical/programmable electronic safety-related systems* [1)]

IEC 61508-4:1998, *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 4: Definitions and abbreviations of terms*

IEC 61508-5:1998, *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 5: Examples of methods for the determination of safety integrity levels*

IEC 61508-6: —, *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 6: Guidelines on the application of parts 2 and 3* [1)]

IEC 61508-7: —, *Functional safety of electrical/electronical/programmable electronic safety-related systems – Part 7: Overview of techniques and measures* [1)]

ISO/IEC Guide 51:1990, *Guidelines for the inclusion of safety aspects in standards*

IEC Guide 104:1997, *Guide to the drafting of safety standards, and the role of Committees with safety pilot functions and safety group functions*

3 Definitions and abbreviations

For the purposes of this standard, the definitions and abbreviations given in IEC 61508-4 apply.

4 Conformance to this standard

The requirements for conformance to this standard are given in clause 4 of IEC 61508-1.

5 Documentation

The objectives and requirements for documentation are given in clause 5 of IEC 61508-1.

---

1) To be published.

6 Système de gestion de la qualité du logiciel

6.1 Objectifs

Les objectifs sont détaillés en 6.1 de la CEI 61508-1.

6.2 Prescriptions

6.2.1 Les prescriptions comprennent celles qui sont détaillées en 6.2 de la CEI 61508-1 ainsi que les prescriptions supplémentaires suivantes.

6.2.2 La planification de sécurité fonctionnelle doit définir la stratégie pour l'approvisionnement, le développement, l'intégration, la vérification, la validation et la modification du logiciel dans les limites requises par le niveau d'intégrité de sécurité du système E/E/PE relatif à la sécurité.

NOTE – La philosophie de cette approche est d'utiliser la planification de sécurité fonctionnelle comme une opportunité d'adapter la présente norme en vue de prendre en compte l'intégrité de sécurité variable requise pour les composants des systèmes E/E/PE relatifs à la sécurité. Il convient de prendre en compte 7.4.2.8 de la partie 3 lorsque des composants de niveau d'intégrité de sécurité différents doivent être utilisés ensemble dans un système E/E/PE relatif à la sécurité.

6.2.3 Il convient que la gestion de configuration du logiciel

a) maîtrise administrativement et techniquement, tout au long du cycle de vie de sécurité du logiciel, la gestion des modifications du logiciel, assurant ainsi la conformité permanente aux prescriptions de sécurité du logiciel spécifiées;

b) garantisse que toutes les opérations nécessaires ont été effectuées en vue de démontrer que le niveau d'intégrité requis de sécurité du logiciel a été atteint;

c) maintienne de manière précise et au moyen d'une identification unique tous les éléments de configuration qui sont nécessaires au maintien de l'intégrité du système E/E/PE relatif à la sécurité. La configuration comprend au moins les éléments suivants: prescriptions et analyse de sécurité, documents de conception et de spécification du logiciel, modules de code source du logiciel, plans de test et résultats, progiciels et composants logiciels préexistants à incorporer au système E/E/PE relatif à la sécurité et tous les outils et environnements de développement qui sont utilisés pour créer, tester ou effectuer une action sur le logiciel du système E/E/PE relatif à la sécurité;

d) applique des procédures de maîtrise des modifications afin d'empêcher toute modification non autorisée; de documenter les demandes de modification; d'analyser l'impact d'une modification proposée, et d'approuver ou rejeter la demande de modification; de documenter les détails et les autorisations pour toutes les modifications approuvées; d'établir le référentiel de la configuration à des points-clés appropriés lors du développement du logiciel, et de documenter le test d'intégration (partielle) qui justifie le référentiel (voir 7.8); de garantir la composition et la construction, de tous les référentiels logiciels (y compris la reconstruction de référentiels précédents);

NOTE 1 – Une autorité de gestion et de décision est nécessaire pour guider et assurer la maîtrise administrative et technique.

e) documente les informations suivantes afin de permettre un audit ultérieur: état de la configuration, état des versions, justification et approbation de toutes les modifications, et détails des modifications;

f) documente de manière formelle la version du logiciel relatif à la sécurité. Il convient de conserver les copies originales du logiciel et toute la documentation associée afin de permettre la maintenance de la modification au cours de l'exploitation de la version du logiciel.

NOTE 2 – Pour tout renseignement complémentaire concernant les processus de gestion de la configuration, voir ISO/CEI 12207.

6 Software quality management system

6.1 Objectives

The objectives are as detailed in 6.1 of IEC 61508-1.

6.2 Requirements

6.2.1 The requirements are as detailed in 6.2 of IEC 61508-1 with the following additional requirements.

6.2.2 The functional safety planning shall define the strategy for the software procurement, development, integration, verification, validation and modification to the extent required by the safety integrity level of the E/E/PE safety related system.

NOTE – The philosophy of this approach is to use the functional safety planning as an opportunity to customise this standard to take account of the varying safety integrity which is required in the E/E/PE safety-related system components. 7.4.2.8 of part 3 should be taken into account when E/E/PE safety-related system components of differing safety integrity levels are to be used together.

6.2.3 Software configuration management should

a) apply administrative and technical controls throughout the software safety lifecycle, in order to manage software changes and thus ensure that the specified requirements for software safety continue to be satisfied;

b) guarantee that all necessary operations have been carried out to demonstrate that the required software safety integrity has been achieved;

c) maintain accurately and with unique identification all configuration items which are necessary to maintain the integrity of the E/E/PE safety-related system. Configuration items include at least the following: safety analysis and requirements; software specification and design documents; software source code modules; test plans and results; pre-existing software components and packages which are to be incorporated into the E/E/PE safety-related system; all tools and development environments which are used to create or test, or carry out any action on, the software of the E/E/PE safety-related system;

d) apply change-control procedures to prevent unauthorized modifications; to document modification requests; to analyse the impact of a proposed modification, and to approve or reject the request; to document the details of, and the authorisation for, all approved modifications; to establish configuration baseline at appropriate points in the software development, and to document the (partial) integration testing which justifies the baseline (see 7.8); to guarantee the composition of, and the building of, all software baselines (including the rebuilding of earlier baselines);

   NOTE 1 – Management decision and authority is needed to guide and enforce the use of administrative and technical controls.

e) document the following information to permit a subsequent audit: configuration status, release status, the justification for and approval of all modifications, and the details of the modification;

f) formally document the release of safety-related software. Master copies of the software and all associated documentation should be kept to permit maintenance and modification throughout the operational lifetime of the released software.

NOTE 2 – For further information on configuration management, see ISO/IEC 12207.

7 Prescriptions concernant le cycle de vie de sécurité du logiciel

7.1 Généralités

7.1.1 Objectif

L'objectif des prescriptions de ce paragraphe est de structurer le développement du logiciel sous forme de phases et d'activités définies (voir tableau 1 et figures 2 à 5).

7.1.2 Prescriptions

7.1.2.1 Un cycle de vie de sécurité pour le développement du logiciel doit être sélectionné et spécifié pendant la planification de sécurité conformément à l'article 6 de la CEI 61508-1.

NOTE – Un modèle de cycle de vie de sécurité conforme aux prescriptions de l'article 7 de la CEI 61508-1 peut être adapté de manière adéquate aux besoins particuliers du projet ou de l'organisation.

7.1.2.2 Les procédures d'assurance qualité et sécurité doivent être intégrées dans les activités du cycle de vie de sécurité.

7.1.2.3 Chaque phase du cycle de vie de sécurité du logiciel doit être divisée en activités élémentaires avec le domaine d'application, les entrées et sorties spécifiées pour chaque phase.

NOTE 1 – Pour tout renseignement complémentaire concernant les phases du cycle de vie, voir ISO/CEI 12207.

NOTE 2 – L'article 5 de la CEI 61508-1 prend en compte les sorties des phases du cycle de vie de sécurité. Durant le développement de certains systèmes E/E/PE relatifs à la sécurité, la sortie de certaines phases du cycle de vie de sécurité peut être couverte par un document distinct tandis que les sorties documentées de plusieurs phases peuvent être fusionnées. La prescription principale est que la sortie de la phase du cycle de vie de sécurité soit adaptée au but prévu. Pour les développements simples, certaines phases du cycle de vie de sécurité peuvent être également fusionnées (voir 7.4.5).

7.1.2.4 A condition que le cycle de vie de sécurité du logiciel satisfasse aux prescriptions de la figure 3 et du tableau 1, il est acceptable d'adapter la profondeur, le nombre et la quantité de travail des phases du modèle en V (voir figure 5), afin de tenir compte de l'intégrité de sécurité et de la complexité du projet.

NOTE – La liste complète des phases du cycle de vie fournie dans le tableau 1 convient pour de grands systèmes nouvellement développés. Pour les petits systèmes, il peut être approprié, par exemple, de fusionner les phases de conception du système logiciel et de conception d'architecture.

7.1.2.5 Il est acceptable d'ordonner le projet logiciel différemment de l'organisation préconisée dans cette norme (c'est-à-dire d'utiliser un autre modèle de cycle de vie de sécurité) à condition que tous les objectifs et prescriptions du présent article soient remplis.

7.1.2.6 Pour chaque phase du cycle de vie, des techniques et mesures appropriées doivent être utilisées. Les annexes A et B (guide pour la sélection de techniques et mesures) donnent des recommandations. Sélectionner des techniques dans les annexes A et B ne garantit pas, de ce fait, que l'intégrité de sécurité requise sera atteinte.

7.1.2.7 Les résultats des activités menées dans le cadre du cycle de vie de sécurité du logiciel doivent être documentés (voir article 5).

7.1.2.8 Si, à un stade quelconque du cycle de vie de sécurité du logiciel, il est nécessaire d'effectuer une modification portant sur une phase précédente du cycle de vie, cette phase précédente du cycle de vie de sécurité doit alors être répétée ainsi que les phases suivantes.

7 Software safety lifecycle requirements

7.1 General

7.1.1 Objective

The objective of the requirements of this subclause is to structure the development of the software into defined phases and activities (see table 1 and figures 2 to 5).

7.1.2 Requirements

7.1.2.1 A safety lifecycle for the development of software shall be selected and specified during safety planning in accordance with clause 6 of IEC 61508-1.

NOTE – A safety lifecycle model which satisfies the requirements of clause 7 of IEC 61508-1 may be suitably customised for the particular needs of the project or organisation.

7.1.2.2 Quality and safety assurance procedures shall be integrated into safety lifecycle activities.

7.1.2.3 Each phase of the software safety lifecycle shall be divided into elementary activities with the scope, inputs and outputs specified for each phase.

NOTE 1 – For further information on lifecycle phases, see ISO/IEC 12207.

NOTE 2 – Clause 5 of IEC 61508-1 considers the outputs from the safety lifecycle phases. In the development of some E/E/PE safety-related systems, the output from some safety lifecycle phases may be a distinct document, while the documented outputs from several phases may be merged. The essential requirement is that the output of the safety lifecycle phase be fit for its intended purpose. In simple developments, some safety lifecycle phases may also be merged (see 7.4.5).

7.1.2.4 Provided that the software safety lifecycle satisfies the requirements of figure 3 and table 1, it is acceptable to tailor the depth, number and work-size of the phases of the V-model (see figure 5) to take account of the safety integrity and the complexity of the project.

NOTE – The full list of lifecycle phases in table 1 is suitable for large newly developed systems. In small systems, it might be appropriate, for example, to merge the phases of software system design and architectural design.

7.1.2.5 It is acceptable to order the software project differently from the organization of this standard (i.e. use another software safety lifecycle model), provided all the objectives and requirements of this clause are met.

7.1.2.6 For each lifecycle phase, appropriate techniques and measures shall be used. Annexes A and B (guide to the selection of techniques and measures) give recommendations. Selecting techniques from annexes A and B does not guarantee by itself that the required safety integrity will be achieved.

7.1.2.7 The results of the activities in the software safety lifecycle shall be documented (see clause 5).

7.1.2.8 If at any stage of the software safety lifecycle, a change is required pertaining to an earlier lifecycle phase, then that earlier safety lifecycle phase and the following phases shall be repeated.

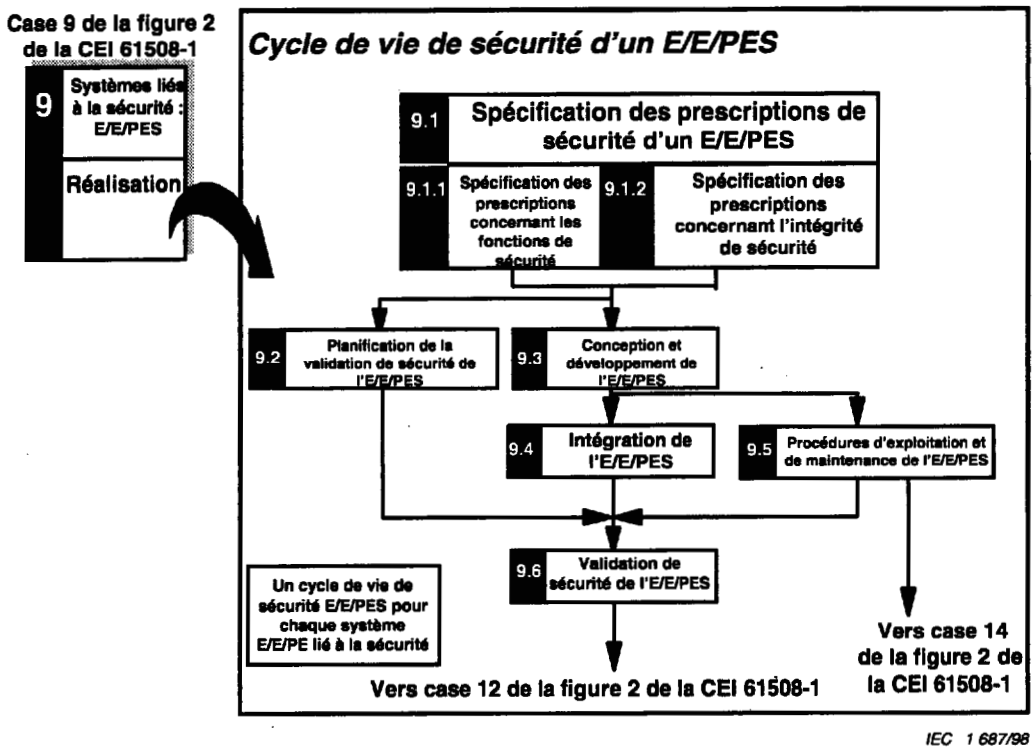


Figure 2 – Cycle de vie de sécurité d'un E/E/PES (en phase de réalisation)

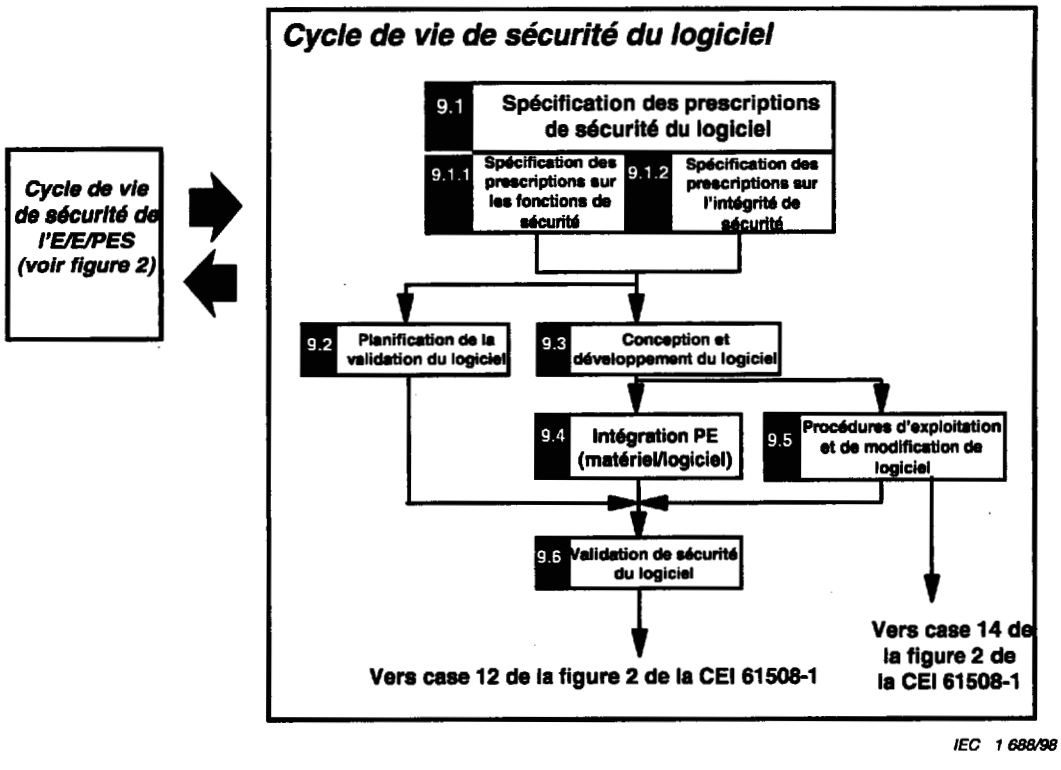


Figure 3 – Cycle de vie de sécurité du logiciel (en phase de réalisation)

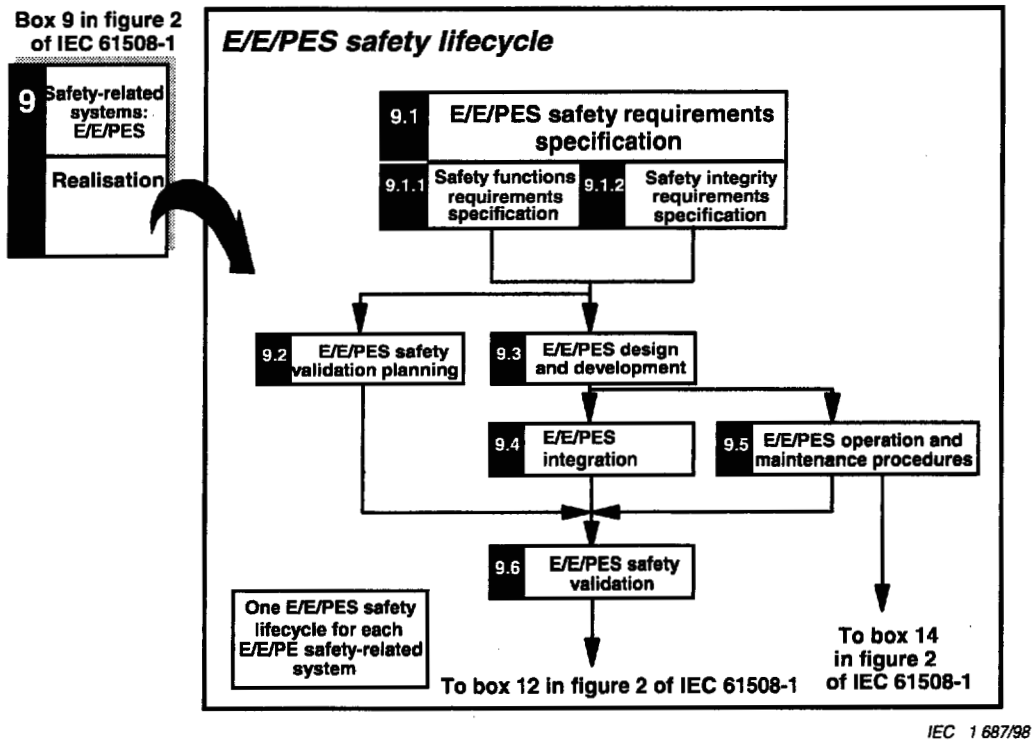


Figure 2 – E/E/PES safety lifecycle (in realisation phase)

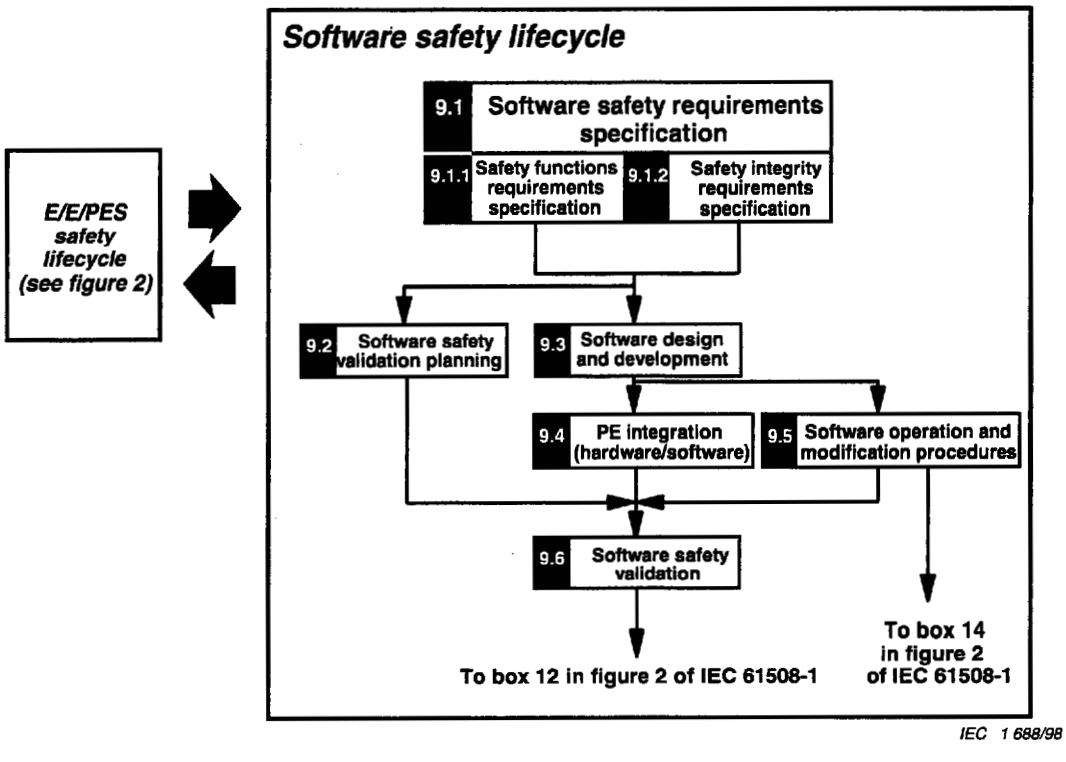


Figure 3 – Software safety lifecycle (in realisation phase)

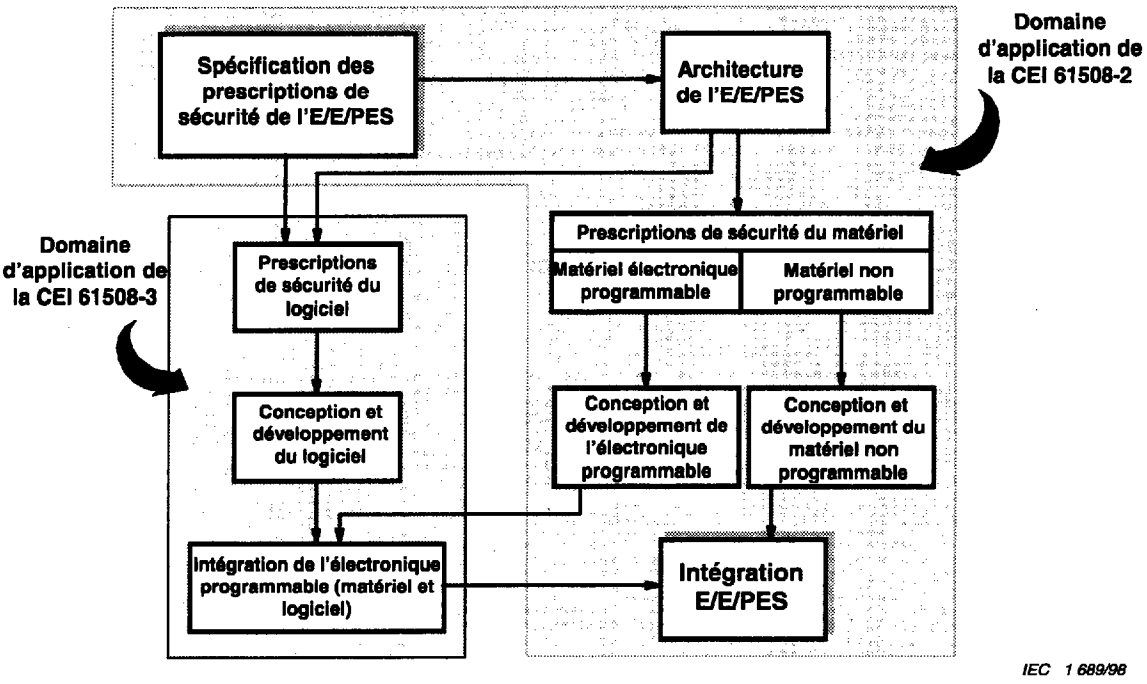


Figure 4 – Relations entre la CEI 61508-2 et la CEI 61508-3 et leurs domaines d'application respectifs

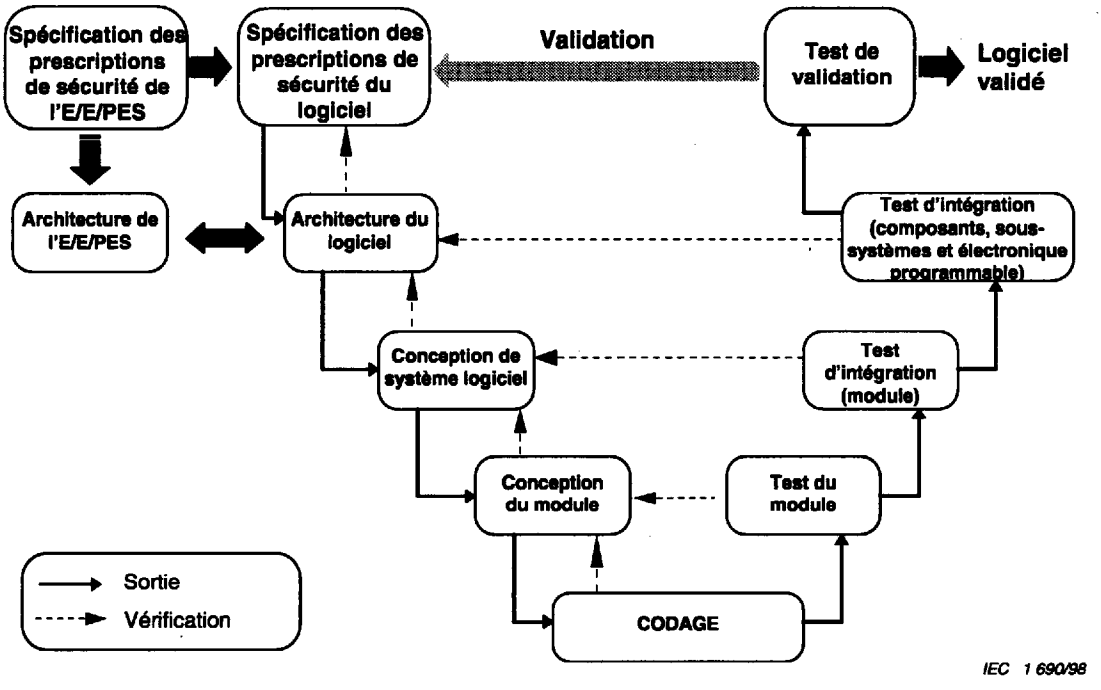


Figure 5 – Intégrité de sécurité du logiciel et cycle de vie de développement (modèle en V)

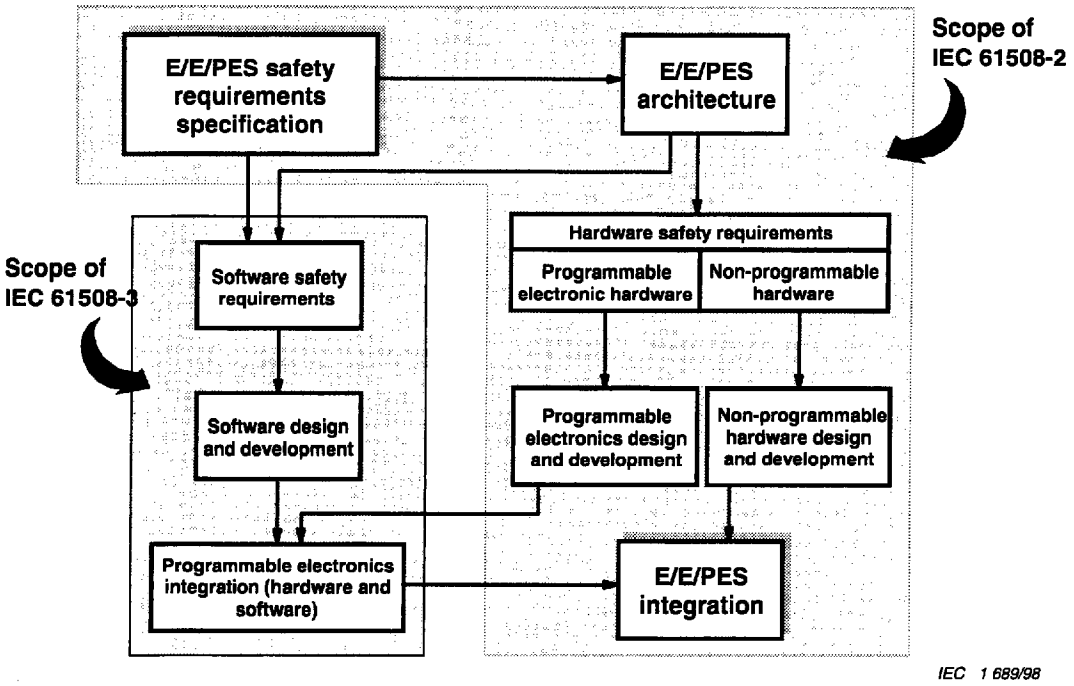


Figure 4 – Relationship between and scope of IEC 6158-2 and IEC 61508-3

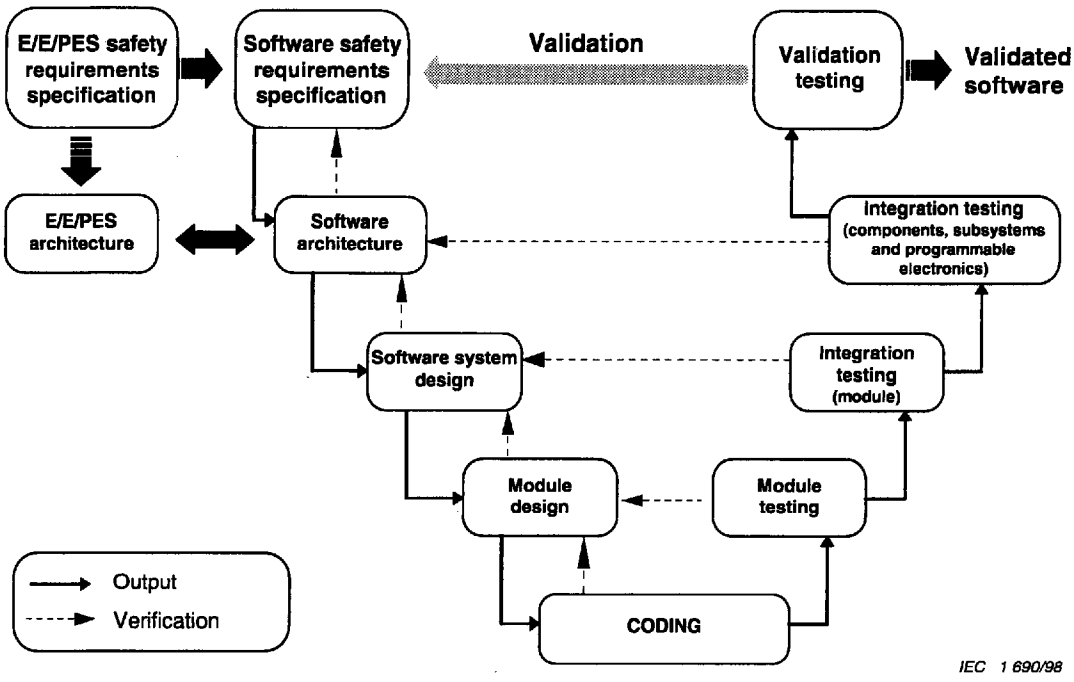


Figure 5 – Software safety integrity and the development lifecycle (the V-model)

Tableau 1 – Cycle de vie de sécurité du logiciel: présentation

Phase du cycle de vie de sécurité		Objectifs	Domaine d'application	Paragraphe des prescriptions	Entrées (informations requises)	Sorties (informations produites)
Numéro de case de la figure 3	Titre					
9.1	Spécification des prescriptions de sécurité du logiciel	Spécifier les prescriptions de sécurité du logiciel concernant les prescriptions des fonctions de sécurité du logiciel et les prescriptions d'intégrité de sécurité du logiciel; spécifier les prescriptions des fonctions de sécurité du logiciel pour chaque système E/E/PE relatif à la sécurité nécessaire à l'implémentation des fonctions de sécurité requises; spécifier les prescriptions d'intégrité de sécurité du logiciel pour chaque système E/E/PE relatif à la sécurité permettant d'atteindre le niveau d'intégrité de sécurité spécifié pour chaque fonction de sécurité qui lui est attribuée.	PES; système logiciel.	7.2.2	Spécification des prescriptions de sécurité des E/E/PES (CEI 61508-2)	Spécification des prescriptions de sécurité du logiciel
9.2	Planification de la validation de sécurité du logiciel	Développer un plan permettant de valider la sécurité du logiciel.	PES; système logiciel.	7.3.2	Spécification des prescriptions de sécurité du logiciel	Plan de validation de sécurité du logiciel
9.3	Conception et développement du logiciel	**Architecture:** Créer une architecture logicielle conforme aux prescriptions spécifiées pour la sécurité du logiciel par rapport au niveau d'intégrité de sécurité requis; passer en revue et évaluer les prescriptions imposées au logiciel par l'architecture matérielle du système E/E/PE relatif à la sécurité, y compris les conséquences des interactions matériel/logiciel sur la sécurité de l'équipement commandé.	PES; système logiciel.	7.4.3	Spécification des prescriptions de sécurité du logiciel; conception de l'architecture du matériel E/E/PES (CEI 61508-2).	Description de la conception de l'architecture du logiciel; spécification du test d'intégration de l'architecture du logiciel; spécification du test d'intégration logiciel / électronique programmable (identique à celui requis dans la CEI 61508-2).
9.3	Conception et développement du logiciel	**Outils supports et langages de programmation:** Sélectionner un ensemble adéquat d'outils d'aide à la vérification, validation, évaluation et modification, y compris les langages et les compilateurs, pour le niveau d'intégrité de sécurité requis au cours du cycle de vie de sécurité complet du logiciel.	PES; système logiciel; outils supports; langage de programmation.	7.4.4	Spécification des prescriptions de sécurité du logiciel; description de la conception de l'architecture du logiciel.	Outils de développement et règles de codage; sélection des outils de développement.

Table 1 – Software safety lifecycle: overview

Safety lifecycle phase		Objectives	Scope	Require-ments subclause	Inputs (information required)	Outputs (information produced)
Figure 3 box number	Title					
9.1	Software safety requirements specification	To specify the requirements for software safety in terms of the requirements for software safety functions and the requirements for software safety integrity; To specify the requirements for the software safety functions for each E/E/PE safety-related system necessary to implement the required safety functions; To specify the requirements for software safety integrity for each E/E/PE safety-related system necessary to achieve the safety integrity level specified for each safety function allocated to that E/E/PE safety-related system.	PES; Software system.	7.2.2	E/E/PES safety requirements specification (IEC 61508-2).	Software safety requirements specification.
9.2	Software safety validation planning	To develop a plan for validating the software safety.	PES; software system.	7.3.2	Software safety requirements specification.	Software safety validation plan.
9.3	Software design and development	**Architecture:** To create a software architecture that fulfils the specified requirements for software safety with respect to the required safety integrity level; To review and evaluate the requirements placed on the software by the hardware architecture of the E/E/PE safety-related system, including the significance of E/E/PE hardware/software interactions for safety of the equipment under control.	PES; software system.	7.4.3	Software safety requirements specification; E/E/PES hardware architecture design (from IEC 61508-2).	Software architecture design description; software architecture integration test specification; software/ programmable electronics integration test specification (the same as IEC 61508-2 requires).
9.3	Software design and development	**Support tools and programming languages:** To select a suitable set of tools, including languages and compilers, for the required safety integrity level, over the whole safety lifecycle of the software which assists verification, validation, assessment and modification.	PES; software system; support tools; program-ming language.	7.4.4	Software safety requirements specification; software architecture design description.	Development tools and coding standards; selection of development tools.

Tableau 1 *(suite)*

Phase du cycle de vie de sécurité		Objectifs	Domaine d'application	Paragraphe des prescriptions	Entrées (informations requises)	Sorties (informations produites)
Numéro de case de la figure 3	Titre					
9.3	Conception et développement du logiciel	**Conception détaillée (conception du système logiciel):** Concevoir et implémenter un logiciel qui réponde aux prescriptions spécifiées pour la sécurité du logiciel conformément au niveau d'intégrité de sécurité requis, qui soit analysable et vérifiable, et qui puisse être modifiable en toute sécurité.	Principaux composants et sous-systèmes de la conception de l'architecture du logiciel.	7.4.5	Description de la conception de l'architecture du logiciel; outils supports et règles de codage.	Spécification de la conception du logiciel; spécification du test d'intégration du système logiciel.
9.3	Conception du logiciel	**Conception détaillée (conception de module logiciel individuel):** Concevoir et implémenter un logiciel qui réponde aux prescriptions spécifiées pour la sécurité du logiciel conformément au niveau d'intégrité de sécurité requis, qui soit analysable et vérifiable, et qui puisse être modifiable en toute sécurité.	Conception du système logiciel.	7.4.5	Spécification de la conception du système logiciel; outils supports et règles de codage.	Spécification de la conception du module logiciel; spécification du test du module logiciel.
9.3	Conception et développement du logiciel	**Implémentation de la conception détaillée:** Concevoir et implémenter un logiciel qui réponde aux prescriptions spécifiées pour la sécurité du logiciel conformément au niveau d'intégrité de sécurité requis, qui soit analysable et vérifiable, et qui puisse être modifiable en toute sécurité.	Modules logiciels individuels	7.4.6	Spécification de conception du module logiciel; outils supports et règles de codage.	Listage des codes sources; rapport de revue de code.
9.3	Conception et développement du logiciel	**Test de module logiciel:** Vérifier que les prescriptions pour la sécurité du logiciel (en termes de fonctions de sécurité logicielles requises et d'intégrité de sécurité du logiciel) ont été remplies pour montrer que chaque module logiciel exécute sa fonction prévue et n'exécute aucune fonction non prévue	Modules logiciels	7.4.7	Spécification de test de module logiciel; listage de code source; rapport de revue de code.	Résultats des tests de module logiciel; modules logiciels vérifiés et testés.
9.3	Conception et développement du logiciel	**Test d'intégration du logiciel:** Vérifier que les prescriptions pour la sécurité du logiciel (en termes de fonctions de sécurité logicielles requises et d'intégrité de sécurité du logiciel) ont été remplies pour montrer que tous les modules logiciels, composants et sous-systèmes interagissent correctement pour exécuter la fonction prévue et n'exécutent aucune fonction non prévue.	Architecture logicielle; système logiciel	7.4.8	Spécification du test d'intégration du système logiciel.	Résultats de test d'intégration du système logiciel; système logiciel vérifié et testé.

Table 1 *(continued)*

Safety lifecycle phase		Objectives	Scope	Require-ments subclause	Inputs (information required)	Outputs (information produced)
Figure 3 box number	Title					
9.3	Software design and development	**Detailed design and development (software system design):** To design and implement software that fulfils the specified requirements for software safety with respect to the required safety integrity level, which is analysable and verifiable, and which is capable of being safely modified.	Major components and subsystems of software architectural design.	7.4.5	Software architecture design description; support tools and coding standards.	Software system design specification; software system integration test specification.
9.3	Software design and development	**Detailed design and development (individual software module design):** To design and implement software that fulfils the specified requirements for software safety with respect to the required safety integrity level, which is analysable and verifiable, and which is capable of being safely modified.	Software system design.	7.4.5	Software system design specification; support tools and coding standards.	Software module design specification; software module test specification.
9.3	Software design and development	**Detailed code implementation:** To design and implement software that fulfils the specified requirements for software safety with respect to the required safety integrity level, which is analysable and verifiable, and which is capable of being safely modified.	Individual software modules.	7.4.6	Software module design specification; support tools and coding standards.	Source code listing; code review report.
9.3	Software design and development	**Software module testing:** To verify that the requirements for software safety (in terms of the required software safety functions and the software safety integrity) have been achieved – to show that each software module performs its intended function and does not perform unintended functions.	Software modules.	7.4.7	Software module test specification; source code listing; code review report.	Software module test results; verified and tested software modules.
9.3	Software design and development	**Software integration testing:** To verify that the requirements for software safety (in terms of the required software safety functions and the software safety integrity) have been achieved – to show that all software modules, components and subsystems interact correctly to perform their intended function and do not perform unintended functions.	Software architecture; software system.	7.4.8	Software system integration test specification.	Software system integration test results; verified and tested software system.

Tableau 1 *(fin)*

Phase du cycle de vie de sécurité		Objectifs	Domaine d'application	Paragraphe des prescriptions	Entrées (informations requises)	Sorties (informations produites)
Numéro de case de la figure 3	Titre					
9.4	Intégration de l'électronique programmable (matériel et logiciel)	Intégrer le logiciel au matériel électronique programmable cible; combiner le logiciel et le matériel dans l'électronique programmable relative à la sécurité pour s'assurer de leur compatibilité et répondre aux prescriptions du niveau d'intégrité de sécurité prévu.	Matériel électronique programmable; logiciel intégré	7.5.2	Spécification du test d'intégration de l'architecture logicielle; spécification du test d'intégration de l'électronique programmable (identique à celle requise dans la CEI 61508-2) électronique programmable intégrée.	Résultats de test d'intégration de l'architecture logicielle; résultats de test d'intégration de l'électronique programmable; électronique programmable intégrée, vérifiée et testée.
9.5	Procédures d'exploitation et de modification du logiciel	Fournir les informations et les procédures concernant le logiciel et permettant d'assurer le maintien de la sécurité fonctionnelle du système E/E/PE relatif à la sécurité durant l'exploitation et les modifications.	Idem ci-dessus	7.6.2	Toutes les entrées ci-dessus, selon le cas.	Procédures d'exploitation et de modification du logiciel.
9.6	Validation de sécurité du logiciel	S'assurer que le système intégré est conforme aux prescriptions spécifiées pour la sécurité du logiciel au niveau d'intégrité de sécurité prévu.	Idem ci-dessus	7.7.2	Plan de validation de sécurité du logiciel.	Résultats de validation de sécurité du logiciel; logiciel validé.
–	Modification du logiciel	Réaliser des corrections, des améliorations ou des adaptations du logiciel validé en s'assurant du maintien du niveau d'intégrité de sécurité du logiciel.	Idem ci-dessus	7.8.2	Procédures de modification du logiciel; demande de modification du logiciel.	Résultats de l'analyse de l'impact de la modification du logiciel; journal de modification du logiciel.
–	Vérification du logiciel	Dans les limites imposées par le niveau d'intégrité de sécurité, tester et évaluer les sorties d'une phase donnée du cycle de vie de sécurité du logiciel afin de vérifier la conformité et la cohérence par rapport aux sorties et aux normes fournies en entrée de cette phase.	Dépend de la phase	7.9.2	Plan de vérification approprié (en fonction de la phase).	Rapport de vérification approprié (en fonction de la phase)..
–	Evaluation de la sécurité fonctionnelle du logiciel	Mener une recherche et parvenir à un jugement sur la sécurité fonctionnelle atteinte par les systèmes E/E/PE relatifs à la sécurité.	Toutes les phases ci-dessus	8	Plan d'évaluation de la sécurité fonctionnelle du logiciel.	Rapport d'évaluation de la sécurité fonctionnelle du logiciel.

Table 1 *(concluded)*

Safety lifecycle phase		Objectives	Scope	Require-ments subclause	Inputs (information required)	Outputs (information produced)
Figure 3 box number	Title					
9.4	Programmable electronics integration (hardware and software)	To integrate the software onto the target programmable electronic hardware; To combine the software and hardware in the safety-related programmable electronics to ensure their compatibility and to meet the requirements of the intended safety integrity level.	Program-mable electronics hardware; integrated software.	7.5.2	Software architecture integration test specification; programmable electronics integration test specification (the same as IEC 61508-2 requires); integrated programmable electronics.	Software architecture integration test results; programmable electronics integration test results; verified and tested integrated programmable electronics.
9.5	Software operation and modification procedures	To provide information and procedures concerning software necessary to ensure that the functional safety of the E/E/PE safety-related system is maintained during operation and modification.	As above	7.6.2	All above, as relevant.	Software operation and modification procedures.
9.6	Software safety validation	To ensure that the integrated system complies with the specified requirements for software safety at the intended safety integrity level.	As above	7.7.2	Software safety validation plan.	Software safety validation results; Validated software.
–	Software modification	To make corrections, enhancements or adaptations to the validated software, ensuring that the required software safety integrity level is sustained.	As above	7.8.2	Software modification procedures; software modification request.	Software modification impact analysis results; software modification log.
–	Software verification	To the extent required by the safety integrity level, to test and evaluate the outputs from a given software safety lifecycle phase to ensure correctness and consis-tency with respect to the outputs and standards provided as input to that phase.	Depends on phase	7.9.2	Appropriate verification plan (depends on phase).	Appropriate verification report (depends on phase).
–	Software functional safety assessment	To investigate and arrive at a judgement on the functional safety achieved by the E/E/PE safety-related systems.	All above phases	8	Software functional safety assessment plan.	Software functional safety assessment report.

7.2 Spécification des prescriptions de sécurité du logiciel

NOTE 1 – Voir aussi les tableaux A.1 et B.7.

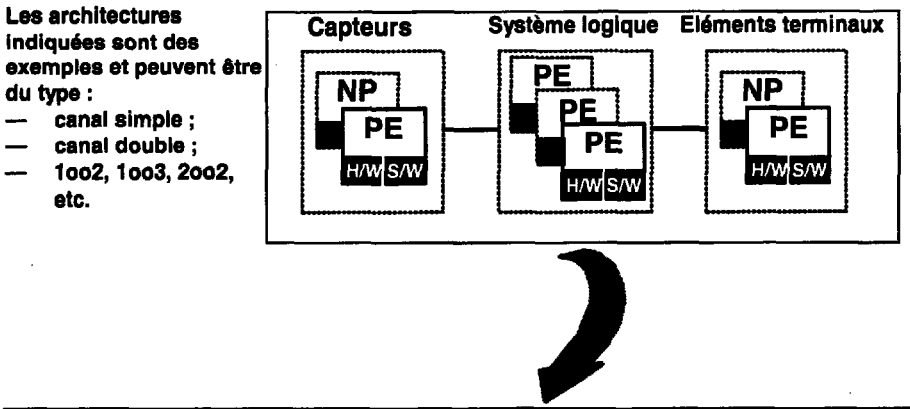
NOTE 2 – Cette phase correspond à la case 9.1 de la figure 3.

7.2.1 Objectifs

7.2.1.1 Le premier objectif des prescriptions de ce paragraphe est de spécifier les prescriptions de sécurité du logiciel comprenant les prescriptions relatives aux fonctions de sécurité du logiciel et les prescriptions relatives à l'intégrité de sécurité du logiciel.

7.2.1.2 Le deuxième objectif des prescriptions de ce paragraphe est de spécifier les prescriptions relatives aux fonctions de sécurité du logiciel pour chaque système E/E/PE relatif à la sécurité nécessaire pour implémenter les fonctions de sécurité requises.

7.2.1.3 Le troisième objectif des prescriptions de ce paragraphe est de spécifier les prescriptions relatives à l'intégrité de sécurité du logiciel pour chaque système E/E/PE relatif à la sécurité permettant d'atteindre le niveau d'intégrité de sécurité spécifié pour chaque fonction de sécurité attribuée à ce système E/E/PE relatif à la sécurité.



Architecture de l'électronique programmable		
Architecture du matériel PE	**Architecture du logiciel PE** (l'architecture logicielle comprend des logiciels embarqués et des logiciels d'application)	
Caractéristiques génériques et spécifiques à l'application concernant la matériel PE. **Exemples :** — **tests de diagnostic ;** — **processeurs redondants ;** — **cartes E/S doublées.**	**Logiciel embarqué PE**	**Logiciel d'application PE**
	Exemples : — **modules de communication ;** — **gestionnaire d'anomalies;** — **exécutifs.**	**Exemples :** — **fonctions d'entrée/sortie ;** — **fonctions dérivées (par exemple vérification des capteurs si non fourni en tant que service du logiciel embarqué).**

IEC 1 691/98

Légende

PE électronique programmable
NP appareil non programmable
H/W matériel
S/W logiciel
MooN M sur N (par exemple 1oo2 est 1 sur 2)

Figure 6 – Relation entre les architectures matérielle et logicielle pour l'électronique programmable

7.2 Software safety requirements specification

NOTE 1 – See also tables A.1 and B.7.

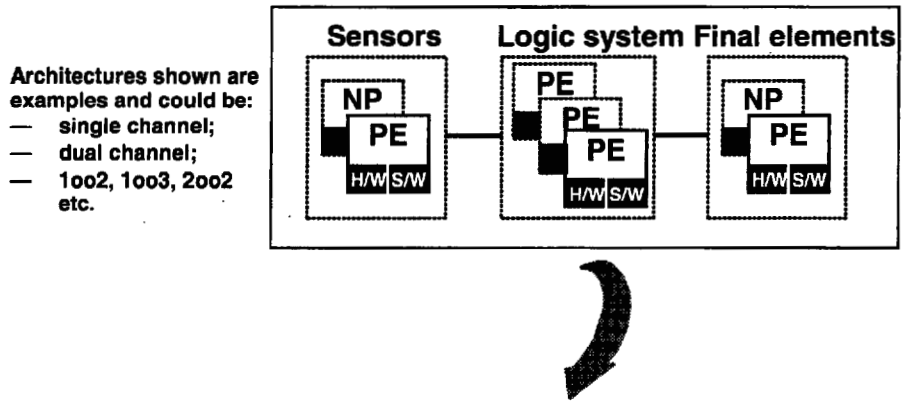
NOTE 2 – This phase is box 9.1 of figure 3.

7.2.1 Objectives

7.2.1.1 The first objective of the requirements of this subclause is to specify the requirements for software safety in terms of the requirements for software safety functions and the requirements for software safety integrity.

7.2.1.2 The second objective of the requirements of this subclause is to specify the requirements for the software safety functions for each E/E/PE safety-related system necessary to implement the required safety functions.

7.2.1.3 The third objective of the requirements of this subclause is to specify the requirements for software safety integrity for each E/E/PE safety-related system necessary to achieve the safety integrity level specified for each safety function allocated to that E/E/PE safety-related system.



Programmable electronics architecture		
PE hardware architecture	**PE software architecture** (s/w architecture consists of embedded s/w and applications s/w)	
Generic and application specific features in PE hardware. Examples include: **— diagnostic tests;** **— redundant processors;** **— dual I/O cards.**	**PE embedded software**	**PE applications software**
	Examples include: **— communications drivers;** **— fault handling;** **— executive software.**	**Examples include:** **— input/output functions;** **— derived functions (for example sensor checking if not provided as a service of the embedded software).**

IEC 1 691/98

Key

PE programmable electronics
NP non-programmable devices
H/W hardware
S/W software
MooN M out of N (for example 1oo2 is 1 out of 2)

Figure 6 — Relationship between the hardware and software architectures of programmable electronics

7.2.2 Prescriptions

NOTE – Ces prescriptions seront satisfaites, dans la plupart des cas, par une combinaison de logiciel générique embarqué et de logiciel spécifique à l'application. C'est la combinaison des deux qui est requise pour fournir les caractéristiques qui satisfont aux paragraphes suivants. La séparation exacte entre logiciel générique et logiciel spécifique à l'application dépend de l'architecture logicielle choisie (voir 7.4.3 et figure 6).

7.2.2.1 Si les prescriptions pour la sécurité du logiciel ont déjà été spécifiées dans les prescriptions pour le système E/E/PE relatif à la sécurité (voir 7.2 de la CEI 61508-2), il n'est pas nécessaire de répéter la spécification des prescriptions de sécurité du logiciel.

7.2.2.2 La spécification des prescriptions relatives à la sécurité du logiciel doit découler des prescriptions de sécurité spécifiées du système E/E/PE relatif à la sécurité (voir CEI 61508-2), et de toute prescription du planning de sécurité (voir article 6). Ces informations doivent être communiquées au développeur du logiciel.

NOTE – Cette prescription ne signifie pas une absence d'itération entre le développeur du système E/E/PES et le développeur du logiciel (CEI 61508-2 et CEI 61508-3). Au fur et à mesure que les prescriptions de sécurité du logiciel et que l'architecture logicielle (voir 7.4.3) se précisent, il est admis qu'il y ait des conséquences sur l'architecture matérielle du système E/E/PE et c'est pourquoi il est essentiel de maintenir une étroite coopération entre les développeurs du matériel et du logiciel. Voir figure 4.

7.2.2.3 La spécification des prescriptions pour la sécurité du logiciel doit être suffisamment détaillée pour permettre d'obtenir l'intégrité de sécurité requise au cours de la conception et de l'implémentation, et permettre une évaluation de la sécurité fonctionnelle.

NOTE – Le niveau de détail de la spécification peut varier en fonction de la complexité de l'application.

7.2.2.4 Le développeur du logiciel doit effectuer une revue des informations contenues en 7.2.2.2 afin de s'assurer que les prescriptions sont correctement spécifiées. Il doit tenir compte en particulier:

a) des fonctions de sécurité;

b) de la configuration ou de l'architecture du système;

c) des prescriptions d'intégrité de sécurité du matériel (électronique programmable, capteurs et actionneurs);

d) des prescriptions d'intégrité de sécurité du logiciel;

e) des performances de capacité et de temps de réponse;

f) des interfaces opérateur et matériel.

7.2.2.5 Le développeur du logiciel doit établir des procédures pour résoudre les divergences concernant l'attribution du niveau d'intégrité de sécurité du logiciel.

7.2.2.6 Dans les limites imposées par le niveau d'intégrité de sécurité, les prescriptions spécifiées pour la sécurité du logiciel doivent être exprimées et structurées de manière à être

a) aussi claires, précises, univoques, vérifiables, testables, maintenables et réalisables que possible en fonction du niveau d'intégrité de sécurité;

b) traçables par rapport à la spécification des prescriptions de sécurité du système E/E/PE relatif à la sécurité;

c) exemptes de toute terminologie ou description ambiguë et/ou susceptible d'être mal comprise par les utilisateurs du document à tous les stades du cycle de vie de sécurité du logiciel.

7.2.2.7 S'ils ne sont pas déjà définis de manière adéquate dans les prescriptions de sécurité du système E/E/PE relatif à la sécurité, tous les modes d'exploitation concernés de l'équipement commandé doivent être détaillés dans les prescriptions spécifiées pour la sécurité du logiciel.

7.2.2 Requirements

NOTE – These requirements will in most cases be achieved by a combination of generic embedded software and application specific software. It is the combination of both that is required to provide the features that satisfy the following subclauses. The exact division between generic and application specific software depends on the chosen software architecture (see 7.4.3 and figure 6).

7.2.2.1 If the requirements for software safety have already been specified in the requirements for the E/E/PE safety-related system (see 7.2 of IEC 61508-2), then the specification of software safety requirements need not be repeated.

7.2.2.2 The specification of the requirements for software safety shall be derived from the specified safety requirements of the E/E/PE safety-related system (see IEC 61508-2), and any requirements of safety planning (see clause 6). This information shall be made available to the software developer.

NOTE – This requirement does not mean that there will be no iteration between the developer of the E/E/PES and the developer of the software (IEC 61508-2 and IEC 61508-3). As the software safety requirements and the software architecture (see 7.4.3) become more precise, there may be an impact on the E/E/PES hardware architecture, and for this reason close co-operation between the hardware and software developer is essential. See figure 4.

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

NOTE – The level of detail of the specification may vary with the complexity of the application.

7.2.2.4 The software developer shall review the information in 7.2.2.2 to ensure that the requirements are adequately specified. In particular the software developer shall consider the following:

a) safety functions;

b) configuration or architecture of the system;

c) hardware safety integrity requirements (programmable electronics, sensors, and actuators);

d) software safety integrity requirements;

e) capacity and response time performance;

f) equipment and operator interfaces.

7.2.2.5 The software developer shall establish procedures for resolving any disagreements over the assignment of the software safety integrity level.

7.2.2.6 To the extent required by the safety integrity level, the specified requirements for software safety shall be expressed and structured so that they are

a) clear, precise, unequivocal, verifiable, testable, maintainable and feasible, commensurate with the safety integrity level;

b) traceable back to the specification of the safety requirements of the E/E/PE safety-related system;

c) free of terminology and descriptions which are ambiguous and/or not understood by those who will utilize the document at any stage of the software safety lifecycle.

7.2.2.7 If not already adequately defined in specified safety requirements of the E/E/PE safety-related system, all relevant modes of operation of the EUC shall be detailed in the specified requirements for software safety.

NOTE – Cette prescription sera remplie dans la plupart des cas par une combinaison de logiciel générique embarqué et de logiciel spécifique à l'application. C'est la combinaison des deux qui est requise pour fournir les caractéristiques qui satisfont aux exigences. La séparation exacte entre logiciel générique et logiciel spécifique à l'application dépend de l'architecture logicielle choisie (voir 7.4.3 et figure 6).

7.2.2.8 La spécification des prescriptions de sécurité du logiciel doit spécifier et documenter toute contrainte entre le matériel et le logiciel, relative à la sécurité ou importante pour celle-ci.

7.2.2.9 Dans les limites imposées par la description de la conception de l'architecture du matériel E/E/PE, la spécification des prescriptions de sécurité du logiciel doit tenir compte

a) de l'auto-surveillance du logiciel (voir exemples en C.2.5 et C.3.10 de la CEI 61508-7);

b) de la surveillance du matériel électronique programmable, des capteurs et des actionneurs;

c) du test périodique des fonctions de sécurité pendant l'exploitation du système;

d) de la possibilité de tester les fonctions de sécurité lorsque l'équipement commandé est en exploitation.

7.2.2.10 Lorsque le système E/E/PE relatif à la sécurité est requis d'exécuter des fonctions qui ne sont pas des fonctions de sécurité, les prescriptions spécifiées pour la sécurité du logiciel doivent alors clairement identifier ces fonctions.

7.2.2.11 La spécification des prescriptions de sécurité du logiciel doit exprimer les propriétés de sécurité requises du produit mais non du projet. En référence aux paragraphes 7.2.2.1 à 7.2.2.10, les points suivants doivent être spécifiés, selon le cas:

a) les prescriptions pour les fonctions de sécurité du logiciel:
   - fonctions permettant à l'équipement commandé d'atteindre ou de maintenir un état sûr;
   - fonctions liées à la détection, signalisation et gestion des défauts du matériel électronique programmable;
   - fonctions liées à la détection, signalisation et gestion des défauts des capteurs et actionneurs;
   - fonctions liées à la détection, signalisation et gestion des défauts du logiciel proprement dit (auto-surveillance du logiciel);
   - fonctions liées au test périodique des fonctions de sécurité en ligne;
   - fonctions liées au test périodique des fonctions de sécurité hors ligne;
   - fonctions permettant la modification sûre du PES;
   - interfaces avec fonctions non relatives à la sécurité;
   - performances de capacité et de temps de réponse;
   - interfaces entres le logiciel et le PES.

     NOTE 1 – Les interfaces comprennent les facilités de programmation en ligne et hors ligne.

b) la prescription pour l'intégrité de sécurité du logiciel:
   - le ou les niveaux d'intégrité de sécurité requis pour chacune des fonctions mentionnées en a) ci-dessus.

     NOTE 2 – Voir l'annexe A de la CEI 61508-5 pour tout renseignement concernant l'allocation de l'intégrité de sécurité aux composants du logiciel.

7.3 Planification de la validation de sécurité du logiciel

NOTE – Cette phase correspond à la case 9.2 de la figure 3.

7.3.1 Objectif

L'objectif des prescriptions du présent paragraphe est d'élaborer un plan pour valider la sécurité du logiciel.

NOTE – This requirement will in most cases be achieved by a combination of generic embedded software and application specific software. It is the combination of both that is required to provide the features that satisfy the requirement. The exact division between generic and application specific software depends on the chosen software architecture (see 7.4.3 and figure 6).

7.2.2.8 The software safety requirements specification shall specify and document any safety-related or relevant constraints between the hardware and the software.

7.2.2.9 To the extent required by the description of the E/E/PE hardware architecture design, the software safety requirements specification shall consider the following:

a) software self-monitoring (for examples see C.2.5 and C.3.10 of IEC 61508-7);

b) monitoring of the programmable electronics hardware, sensors, and actuators;

c) periodic testing of safety functions while the system is running;

d) enabling safety functions to be testable when the EUC is operational.

7.2.2.10 When the E/E/PE safety-related system is required to perform non-safety functions, then the specified requirements for software safety shall clearly identify these functions.

7.2.2.11 The software safety requirements specification shall express the required safety properties of the product, but not of the project. With reference to 7.2.2.1 to 7.2.2.10, the following shall be specified as appropriate:

a) the requirements for the software safety functions:
   - functions that enable the EUC to achieve or maintain a safe state;
   - functions related to the detection, annunciation and management of faults in the programmable electronics hardware;
   - functions related to the detection, annunciation and management of sensor and actuators faults;
   - functions related to the detection, annunciation and management of faults in the software itself (software self-monitoring);
   - functions related to the periodic testing of safety functions on-line;
   - functions related to the periodic testing of safety functions off-line;
   - functions that allow the PES to be safely modified;
   - interfaces to non safety-related functions;
   - capacity and response time performance;
   - interfaces between the software and the PES.

     NOTE 1 – Interfaces include both off-line and online programming facilities.

b) the requirement for the software safety integrity:
   - the safety integrity level(s) for each of the functions in a) above.

     NOTE 2 – See annex A of IEC 61508-5 for information concerning the allocation of safety integrity to software components.

7.3 Software safety validation planning

NOTE – This phase is box 9.2 of figure 3.

7.3.1 Objective

The objective of the requirements of this subclause is to develop a plan for validating the software safety.

7.3.2 Prescriptions

7.3.2.1 La planification doit être conduite afin de spécifier les étapes, qu'elles soient procédurales ou techniques, qui seront utilisées pour démontrer que le logiciel satisfait aux prescriptions de sécurité (voir 7.2).

7.3.2.2 Le plan pour valider la sécurité du logiciel doit tenir compte

a) des détails sur quand la validation aura lieu;

b) des détails concernant les personnes qui doivent effectuer la validation;

c) de l'identification des modes d'exploitation concernés de l'équipement commandé, comprenant:
   - les préparatifs d'utilisation, y compris l'initialisation et le réglage;
   - le démarrage, l'apprentissage, le mode automatique, le mode manuel, le mode semi-automatique et le fonctionnement en état stable;
   - la remise à zéro, l'arrêt et la maintenance;
   - les conditions anormales raisonnablement prévisibles;

d) l'identification du logiciel relatif à la sécurité nécessitant une validation pour chaque mode d'exploitation de l'équipement commandé avant le début de la mise en service;

e) la stratégie technique de la validation (par exemple les méthodes analytiques, les tests statistiques, etc.) (voir 7.3.2.3);

f) en accord avec le point e), les mesures (techniques) et les procédures qui doivent être utilisées pour confirmer que chaque fonction de sécurité est conforme aux prescriptions spécifiées pour les fonctions de sécurité du logiciel (voir 7.2) et aux prescriptions spécifiées pour l'intégrité de sécurité du logiciel (voir 7.2);

g) la référence spécifique aux prescriptions spécifiées pour la sécurité du logiciel (voir 7.2);

h) l'environnement requis dans lequel les activités de validation doivent se dérouler (par exemple, pour des tests, cet environnement comprendrait les outils étalonnés ainsi que les équipements de test);

i) les critères d'acceptation ou de rejet (voir 7.3.2.5);

j) les politiques et procédures d'évaluation des résultats de la validation, en particulier en ce qui concerne les résultats conduisant à des rejets.

NOTE – Ces prescriptions sont basées sur les prescriptions générales décrites en 7.8 de la CEI 61508-1.

7.3.2.3 La stratégie technique concernant la validation d'un logiciel relatif à la sécurité (voir tableau A.7) doit comprendre les informations suivantes:

a) le choix entre des techniques manuelles ou automatiques, ou les deux;

b) le choix entre des techniques statiques ou dynamiques, ou les deux;

c) le choix entre des techniques analytiques ou statistiques, ou les deux.

7.3.2.4 Dans le cadre de la procédure de validation d'un logiciel relatif à la sécurité, le domaine d'application et le contenu de la planification pour valider la sécurité du logiciel doivent être passés en revue avec l'évaluateur ou un représentant de l'évaluateur, si cela est requis par le niveau d'intégrité de sécurité (voir 8.2.12 de la CEI 61508-1). Cette procédure doit également contenir une clause concernant la présence de l'évaluateur durant le test.

7.3.2.5 Les critères d'acceptation ou de rejet du test de validation du logiciel doivent comprendre

a) les signaux d'entrée requis avec leurs séquences et leurs valeurs;

7.3.2 Requirements

7.3.2.1 Planning shall be carried out to specify the steps, both procedural and technical, that will be used to demonstrate that the software satisfies its safety requirements (see 7.2).

7.3.2.2 The plan for validating the software safety shall consider the following:

a) details of when the validation shall take place;

b) details of those who shall carry out the validation;

c) identification of the relevant modes of the EUC operation including
   - preparation for use including setting and adjustment;
   - start up; teach; automatic; manual; semi-automatic; steady state of operation;
   - re-setting; shut down; maintenance;
   - reasonably foreseeable abnormal conditions;

d) identification of the safety-related software which needs to be validated for each mode of EUC operation before commissioning commences;

e) the technical strategy for the validation (for example analytical methods, statistical tests etc.) (see 7.3.2.3);

f) In accordance with item e), the measures (techniques) and procedures that shall be used for confirming that each safety function conforms with the specified requirements for the software safety functions (see 7.2), and the specified requirements for software safety integrity (see 7.2);

g) specific reference to the specified requirements for software safety (see 7.2);

h) the required environment in which the validation activities are to take place (for example for tests this would include calibrated tools and equipment);

i) the pass/fail criteria (see 7.3.2.5);

j) the policies and procedures for evaluating the results of the validation, particularly failures.

NOTE – These requirements are based on the general requirements of 7.8 of IEC 61508-1.

7.3.2.3 The technical strategy for the validation of safety-related software (see table A.7) shall include the following information:

a) choice of manual or automated techniques or both;

b) choice of static or dynamic techniques or both;

b) choice of analytical or statistical techniques or both.

7.3.2.4 As part of the procedure for validating safety-related software, the scope and contents of the planning for validating the software safety shall be reviewed with the assessor or with a party representing the assessor, if required by the safety integrity level (see 8.2.12 of IEC 61508-1). This procedure shall also make a statement concerning the presence of the assessor during testing.

7.3.2.5 The pass/fail criteria for accomplishing software validation shall include

a) the required input signals with their sequences and their values;

b) les signaux de sortie prévus avec leurs séquences et leurs valeurs;

c) les autres critères d'acceptation, tels que l'utilisation de la mémoire, la synchronisation et les tolérances applicables aux valeurs.

7.4 Conception et développement du logiciel

NOTE – Cette phase correspond à la case 9.3 de la figure 3.

7.4.1 Objectifs

7.4.1.1 Le premier objectif des prescriptions du présent paragraphe est de créer une architecture de logiciel satisfaisant aux prescriptions spécifiées pour la sécurité du logiciel (voir 7.2) en fonction du niveau d'intégrité de sécurité requis.

7.4.1.2 Le second objectif des prescriptions du présent paragraphe est de passer en revue et d'évaluer les prescriptions imposées au logiciel par l'architecture matérielle du système E/E/PE relatif à la sécurité, y compris les conséquences des interactions matériel/logiciel de l'E/E/PE sur la sécurité de l'équipement commandé.

7.4.1.3 Le troisième objectif des prescriptions du présent paragraphe est de sélectionner un ensemble adéquat d'outils d'aide à la vérification, validation, évaluation et modification, y compris les langages et les compilateurs, pour le niveau d'intégrité de sécurité requis au cours du cycle de vie de sécurité complet du logiciel.

7.4.1.4 Le quatrième objectif des prescriptions du présent paragraphe est de concevoir et implémenter un logiciel qui réponde aux prescriptions spécifiées pour la sécurité du logiciel (voir 7.2) en fonction du niveau d'intégrité de sécurité requis, qui soit analysable et vérifiable, et qui soit modifiable de façon sûre.

7.4.1.5 Le cinquième objectif des prescriptions du présent paragraphe est de vérifier que les prescriptions de sécurité du logiciel (en termes de fonctions de sécurité du logiciel requises et d'intégrité de sécurité du logiciel) ont été remplies.

7.4.2 Prescriptions générales

7.4.2.1 En fonction de la nature du développement du logiciel, la responsabilité en ce qui concerne le respect des prescriptions mentionnées en 7.4 peut incomber au fournisseur seul, à l'utilisateur seul, ou aux deux ensemble. L'attribution des responsabilités doit être déterminée pendant la planification de sécurité (voir article 6).

7.4.2.2 Conformément au niveau d'intégrité de sécurité requis, la méthode de conception choisie doit posséder des caractéristiques qui facilitent

a) l'abstraction, la modularité et autres caractéristiques pour maîtriser la complexité;

b) l'expression
   - de la fonctionnalité,
   - du flot d'informations entre les composants,
   - du séquencement et des informations liées au temps,
   - des contraintes de temps,
   - des conflits d'accès,
   - des structures de données et de leurs propriétés,
   - des hypothèses de conception et de leurs liens;

c) la compréhension par les développeurs et tous ceux qui ont besoin de comprendre la conception;

d) la vérification et la validation.

NOTE – Voir aussi les tableaux en annexes A et B.

b) the anticipated output signals with their sequences and their values; and

c) other acceptance criteria, for example memory usage, timing and value tolerances.

7.4 Software design and development

NOTE – This phase is box 9.3 of figure 3.

7.4.1 Objectives

7.4.1.1 The first objective of the requirements of this subclause is to create a software architecture that fulfils the specified requirements for software safety (see 7.2) with respect to the required safety integrity level.

7.4.1.2 The second objective of the requirements of this subclause is to review and evaluate the requirements placed on the software by the hardware architecture of the E/E/PE safety-related system, including the significance of E/E/PE hardware/software interactions for safety of the equipment under control.

7.4.1.3 The third objective of the requirements of this subclause is to select a suitable set of tools, including languages and compilers, for the required safety integrity level, over the whole safety lifecycle of the software which assists verification, validation, assessment and modification.

7.4.1.4 The fourth objective of the requirements of this subclause is to design and implement software that fulfils the specified requirements for software safety (see 7.2) with respect to the required safety integrity level, which is analysable and verifiable, and which is capable of being safely modified.

7.4.1.5 The fifth objective of the requirements of this subclause is to verify that the requirements for software safety (in terms of the required software safety functions and the software safety integrity) have been achieved.

7.4.2 General requirements

7.4.2.1 Depending on the nature of the software development, responsibility for conformance with 7.4 can rest with the supplier alone, or with the user alone, or with both. The division of responsibility shall be determined during safety planning (see clause 6).

7.4.2.2 In accordance with the required safety integrity level, the design method chosen shall possess features that facilitate:

a) abstraction, modularity and other features which control complexity;

b) the expression of:
   - functionality,
   - information flow between components,
   - sequencing and time related information,
   - timing constraints,
   - concurrency,
   - data structures and their properties,
   - design assumptions and their dependencies;

c) comprehension by developers and others who need to understand the design;

d) verification and validation.

NOTE – See also the tables in annexes A and B.

7.4.2.3 La testabilité et l'aptitude à la modification sûre doivent être prises en compte durant les activités de conception afin de faciliter la mise en oeuvre de ces propriétés dans le système final relatif à la sécurité.

NOTE – A titre d'exemple, on peut citer les modes de maintenance dans les industries de machines-outils et de procédé.

7.4.2.4 La méthode de conception choisie doit avoir des caractéristiques qui facilitent la modification du logiciel. Ces caractéristiques permettent par exemple la modularité, le masquage d'informations et l'encapsulation.

7.4.2.5 Les représentations de la conception doivent être basées sur une notation clairement définie ou limitée à des caractéristiques clairement définies.

7.4.2.6 Dans la mesure du possible, la conception doit minimiser la partie du logiciel relative à la sécurité.

7.4.2.7 Lorsque le logiciel doit implémenter à la fois des fonctions de sécurité et des fonctions qui ne sont pas de sécurité, le logiciel dans son ensemble doit alors être considéré comme relatif à la sécurité, à moins qu'il soit possible de démontrer au niveau de la conception l'indépendance des fonctions de manière appropriée.

7.4.2.8 Lorsque le logiciel doit implémenter des fonctions de sécurité correspondant à différents niveaux d'intégrité de sécurité, le logiciel dans son ensemble doit alors être considéré comme appartenant au niveau d'intégrité de sécurité le plus élevé, à moins qu'une indépendance suffisante des fonctions de sécurité correspondant aux différents niveaux d'intégrité de sécurité soit démontrée au niveau de la conception. La justification de l'indépendance des fonctions doit être documentée.

NOTE – Il faut que le niveau d'intégrité de sécurité du logiciel soit au moins aussi élevé que le niveau d'intégrité de sécurité de la fonction de sécurité à laquelle il appartient. Toutefois, le niveau d'intégrité de sécurité d'un composant logiciel peut être inférieur au niveau d'intégrité de sécurité de la fonction de sécurité à laquelle le composant logiciel appartient si le composant est combiné avec d'autres composants matériels, de manière que le niveau d'intégrité de sécurité de la combinaison soit au moins égal à celui de la fonction de sécurité.

7.4.2.9 Dans la mesure du possible, la conception doit comprendre des fonctions logicielles pour exécuter des essais périodiques et tous les essais de diagnostic de façon à satisfaire à la prescription d'intégrité de sécurité du système E/E/PE relatif à la sécurité (ainsi que cela est exposé dans la CEI 61508-2).

7.4.2.10 La conception du logiciel doit comprendre, en fonction du niveau d'intégrité de sécurité requis, une auto-surveillance des flots de contrôle et de données. En cas de détection d'une défaillance, les mesures appropriées doivent être prises. Voir tableaux A.2 et A.4.

7.4.2.11 Si un logiciel standard ou développé précédemment doit être utilisé pour la conception (voir tableaux A.3 et A.4), il doit être alors clairement identifié. La capacité du logiciel à satisfaire à la spécification des prescriptions de sécurité du logiciel (voir 7.2) doit être justifiée. Cette capacité du logiciel doit être établie à partir de la constatation d'un fonctionnement satisfaisant dans une application similaire, ou de l'application des mêmes procédures de vérification et de validation que celles auxquelles sont normalement soumis les logiciels développés qui sont nouveaux. Il convient en outre d'évaluer les contraintes imposées par l'environnement du logiciel précédent (par exemple les dépendances au système d'exploitation et au compilateur).

NOTE – La justification peut être réalisée pendant la planification de sécurité (voir article 6).

7.4.2.12 Le présent paragraphe (7.4) doit, dans la mesure du possible, s'appliquer à toutes les données, y compris les langages de génération des données.

7.4.2.3 Testability and the capacity for safe modification shall be considered during the design activities in order to facilitate implementation of these properties in the final safety-related system.

NOTE – Examples include maintenance modes in machinery and process plant.

7.4.2.4 The design method chosen shall possess features that facilitate software modification. Such features include modularity, information hiding and encapsulation.

7.4.2.5 The design representations shall be based on a notation which is unambiguously defined or restricted to unambiguously defined features.

7.4.2.6 As far as practicable the design shall minimise the safety-related part of the software.

7.4.2.7 Where the software is to implement both safety and non-safety functions, then all of the software shall be treated as safety-related, unless adequate independence between the functions can be demonstrated in the design.

7.4.2.8 Where the software is to implement safety functions of different safety integrity levels, then all of the software shall be treated as belonging to the highest safety integrity level, unless adequate independence between the safety functions of the different safety integrity levels can be shown in the design. The justification for independence shall be documented.

NOTE – The software safety integrity level must be at least as high as the safety integrity level of the safety function to which it belongs. However, the safety integrity level of a software component can be lower than the safety integrity level of the the safety function to which the software component belongs, if the component is used in combination with other hardware components such that the safety integrity level of the combination at least equals that of the the safety function.

7.4.2.9 As far as practicable, the design shall include software functions to execute proof tests and all diagnostic tests in order to fulfil the safety integrity requirement of the E/E/PE safety-related system (as set out in IEC 61508-2).

7.4.2.10 The software design shall include, commensurate with the required safety integrity level, self-monitoring of control flow and data flow. On failure detection, appropriate actions shall be taken. See table A.2 and table A.4.

7.4.2.11 If standard or previously developed software is to be used as part of the design (see tables A.3 and A.4) then it shall be clearly identified. The software suitability in satisfying the specification of requirements for software safety (see 7.2) shall be justified. Suitability shall be based upon evidence of satisfactory operation in a similar application or having been subject to the same verification and validation procedures as would be expected for any newly developed software. Constraints from the previous software environment (for example operating system and compiler dependencies) should be evaluated.

NOTE – The justification may be developed during safety planning (see clause 6).

7.4.2.12 This subclause (7.4) shall, in so far as it is appropriate, apply to data including any data generation languages.

7.4.3 Prescriptions concernant l'architecture du logiciel

NOTE 1 – Voir aussi tableaux A.2 et B.7.

NOTE 2 – L'architecture du logiciel définit les principaux composants et sous-systèmes du logiciel, comment ils sont interconnectés et comment les attributs requis, en particulier l'intégrité de sécurité, sont obtenus. Les principaux composants du logiciel comprennent normalement les systèmes d'exploitation, les bases de données, les sous-systèmes d'entrée/sortie de l'installation, les sous-systèmes de communication, le ou les programmes d'application, les outils de programmation et de diagnostic, etc.

NOTE 3 – Dans certains secteurs industriels, l'architecture du logiciel est appelée «description fonctionnelle» ou «spécification de conception fonctionnelle» (bien que ces documents puissent également couvrir le matériel).

NOTE 4 – Pour la programmation de l'application utilisateur dans les langages de variabilité limitée, particulièrement dans les PLC (voir annexe E de la CEI 61508-6), l'architecture est donnée par le fournisseur comme une caractéristique standard du PLC. Le fournisseur serait alors requis, dans le cadre de la présente norme, de garantir à l'utilisateur la conformité de ses produits avec les prescriptions mentionnées en 7.4. L'utilisateur adapte le PLC à son application en programmant de façon standard, par exemple dans le langage à contact. Les prescriptions de 7.4.3 à 7.4.8 sont aussi applicables. La prescription de définir et documenter l'architecture peut être considérée comme une information utilisable par l'utilisateur pour sélectionner le PLC (ou équivalent) pour son application.

NOTE 5 – A l'autre extrême, dans certaines applications embarquées utilisant un langage à variabilité totale comme, par exemple, une machine commandée par microprocesseur, il faudra que le fournisseur crée l'architecture spécialement pour cette application (ou classe d'application). L'utilisateur ne dispose normalement pas de moyen de programmation. Dans ce cas, c'est le fournisseur qui est responsable de la conformité de son produit avec les prescriptions mentionnées en 7.4.

NOTE 6 – Certains systèmes appartiennent aux deux types de systèmes mentionnés dans les notes 4 et 5. L'utilisateur et le fournisseur se partagent en conséquence la responsabilité de la conformité.

NOTE 7 – Du point de vue de la sécurité, c'est pendant la phase d'architecture du logiciel qu'est développée la stratégie de base concernant la sécurité pour le logiciel.

7.4.3.1 Suivant le type de développement du logiciel, la responsabilité en ce qui concerne la conformité avec les prescriptions mentionnées en 7.4.3 peut incomber au fournisseur seul, à l'utilisateur seul, ou aux deux ensemble (voir les notes ci-dessus). La répartition de la responsabilité doit être documentée pendant la planification de sécurité (voir article 6).

7.4.3.2 La conception de l'architecture logicielle proposée doit être définie par le fournisseur et/ou le développeur et une description détaillée de cette conception doit être fournie. La description doit

a) sélectionner et justifier un ensemble intégré de techniques et de mesures nécessaires pendant les phases du cycle de vie de sécurité du logiciel pour satisfaire à la spécification des prescriptions de sécurité du logiciel en fonction du niveau d'intégrité de sécurité requis (voir 7.2). Ces techniques et mesures comprennent les stratégies de conception du logiciel pour la tolérance aux anomalies (cohérente avec le matériel) et l'évitement des anomalies, incluant (lorsque cela est approprié) la redondance et la diversité;

b) être basée sur une partition en sous-systèmes/composants, avec les informations suivantes fournies pour chacun d'entre eux:

   - sous-système/composant nouveau, existant ou propriétaire;
   - sous-système/composant vérifié précédemment ou non et, si oui, conditions de la vérification;
   - sous-système/composant relatif ou non à la sécurité;
   - niveau d'intégrité de sécurité du sous-système/composant;

c) déterminer toutes les interactions matériel/logiciel, et évaluer et détailler leur importance;

d) utiliser une notation pour représenter l'architecture définie de façon non ambiguë ou limitée à des caractéristiques définies de façon non ambiguë;

e) sélectionner les caractéristiques d'architecture à utiliser pour maintenir l'intégrité de sécurité de toutes les données. Il est admis que ces données comprennent les données d'entrée/sortie de l'installation, les données de communication, les données d'interface opérateur, les données de maintenance et celles de la base de données interne;

f) spécifier les tests d'intégration de l'architecture logicielle appropriés pour s'assurer que cette architecture satisfait à la spécification des prescriptions pour la sécurité du logiciel en fonction du niveau d'intégrité de sécurité requis (voir 7.2).

7.4.3 Requirements for software architecture

NOTE 1 – See also tables A.2 and B.7.

NOTE 2 – The software architecture defines the major components and subsystems of the software, how they are interconnected, and how the required attributes, particularly safety integrity, will be achieved. Examples of major software components include operating systems, databases, plant input/output subsystems, communication subsystems, application program(s), programming and diagnostic tools etc.

NOTE 3 – In certain industrial sectors the software architecture would be called a function description or functional design specification (although these documents could also include the hardware).

NOTE 4 – For user application programming in limited variability languages particularly in PLCs (see annex E of IEC 61508-6) the architecture is provided by the supplier as a standard feature of the PLC. The supplier would, under this standard, be required to assure the user of the compliance of his products to the requirements of 7.4. The user tailors the PLC to the application by using the standard programming facilities, for example ladder logic. The requirements of 7.4.3 to 7.4.8 still apply. The requirement to define and document the architecture can be seen as information that the user would use to select the PLC (or equivalent) for the application.

NOTE 5 – At the other extreme, in certain embedded applications using a full variability language, for example a microprocessor controlled machine, the architecture will need to be created especially for the application (or class of application) by the supplier. The user usually has no programming facilities. In these circumstances, responsibility for assuring compliance with 7.4 rests with the supplier.

NOTE 6 – There are systems falling in between the two types of systems mentioned in notes 4 and 5, and responsibility for compliance will therefore be shared between the user and the supplier.

NOTE 7 – From a safety viewpoint, the software architecture phase is where the basic safety strategy is developed for the software.

7.4.3.1 Depending on the nature of the software development, responsibility for conformance with 7.4.3 can rest with the supplier alone, or with the user alone, or with both (see notes above). The division of responsibility shall be documented during safety planning (see clause 6).

7.4.3.2 The proposed software architecture design shall be established by the software supplier and/or developer, and a description of the software architecture design shall be detailed. The description shall

a) select and justify an integrated set of techniques and measures necessary during the software safety lifecycle phases to satisfy the specification of requirements for software safety at the required safety integrity level (see 7.2). These techniques and measures include software design strategies for both fault tolerance (consistent with the hardware) and fault avoidance, including (where appropriate) redundancy and diversity;

b) be based on a partitioning into components/subsystems, for each of which the following information shall be provided:

   – whether they are new, existing or proprietary;

   – whether they have been previously verified, and if yes, their verification conditions;

   – whether each subsystem/component is safety-related or not;

   – the software safety integrity level of the subsystem/component;

c) determine all software/hardware interactions and evaluate and detail their significance;

d) use a notation to represent the architecture which is unambiguously defined or restricted to unambiguously defined features;

e) select the design features to be used for maintaining the safety integrity of all data. Such data may include plant input-output data, communications data, operator interface data, maintenance data and internal database data;

f) specify appropriate software architecture integration tests to ensure that the software architecture satisfies the specification of requirements for software safety at the required safety integrity level (see 7.2).

7.4.3.3 Toute modification des prescriptions de sécurité spécifiées du système E/E/PE relatif à la sécurité requise après l'application du paragraphe 7.4.3.2 doit recevoir l'accord du développeur du système E/E/PE et être documentée.

NOTE – Il y aura inévitablement itération entre l'architecture matérielle et l'architecture logicielle (voir figure 5). Il est donc nécessaire de discuter avec le développeur du matériel les problèmes concernant, par exemple, la spécification de test pour l'intégration du logiciel et du matériel de l'électronique programmable (voir 7.5).

7.4.4 Prescriptions concernant les outils supports et les langages de programmation

NOTE 1 – Voir aussi tableau A.3.

NOTE 2 – Le choix des outils de développement dépendra de la nature des activités de développement du logiciel et de l'architecture du logiciel (voir 7.4.3).

- Pour la programmation de l'application utilisateur dans un langage à variabilité limitée à de faibles niveaux d'intégrité de sécurité, il est permis que les langages de programmation et les outils requis se limitent aux langages PLC standard et aux programmes d'édition et de chargement. La responsabilité du respect du paragraphe 7.4.4 incombera donc essentiellement au fournisseur.
- Pour des niveaux d'intégrité de sécurité supérieurs, il est admis que l'utilisation de sous-ensembles limités de langage PLC soit nécessaire et que celle d'outils de vérification et de validation tels que les analyseurs de code et les simulateurs soit nécessaire. Dans ce cas, la responsabilité incombe à la fois au fournisseur et à l'utilisateur.
- Les outils pour les applications embarquées utilisant un langage de variabilité totale devront être nécessairement plus détaillés, même pour des niveaux d'intégrité de sécurité inférieurs. La responsabilité du respect du paragraphe 7.4.4 incombera essentiellement au développeur du logiciel. Cela s'applique au fournisseur de langage PLC qui utiliserait des langages de variabilité totale dans l'élaboration du langage de variabilité limitée pour la programmation des applications utilisateur.

7.4.4.1 Suivant la nature du développement du logiciel, la responsabilité concernant le respect des prescriptions mentionnées en 7.4.4 peut incomber au fournisseur seul, à l'utilisateur seul, ou aux deux (voir note 2 ci-dessus). La répartition de la responsabilité doit être documentée pendant la planification de sécurité (voir article 6).

7.4.4.2 Un ensemble adapté d'outils intégrés comprenant des langages, des compilateurs, des outils de gestion de configuration et, quand c'est applicable, des outils de test automatique doit être sélectionné en fonction du niveau d'intégrité de sécurité requis. La capacité des outils adaptés (pas nécessairement ceux utilisés pendant le développement initial du système) à fournir les services concernés pendant toute la durée de vie du système E/E/PE relatif à la sécurité doit être prise en compte.

7.4.4.3 Dans les limites prescrites par le niveau d'intégrité de sécurité, le langage de programmation sélectionné doit:

a) comprendre un traducteur/compilateur qui, soit est couvert par un certificat de validation conforme à une norme nationale ou internationale reconnue, soit doit être évalué en vue d'établir son aptitude à l'utilisation prévue;

b) être totalement défini de façon non ambiguë, ou limité à des caractéristiques définies de façon non ambiguë;

c) correspondre aux caractéristiques de l'application;

d) comporter des caractéristiques qui facilitent la détection des erreurs de programmation;

e) supporter des caractéristiques adaptées à la méthode de conception.

7.4.4.4 S'il n'est pas possible de satisfaire aux prescriptions mentionnées en 7.4.4.3, la justification de tout langage de remplacement utilisé doit être documentée pendant la description de la conception de l'architecture du logiciel (voir 7.4.3). La justification doit détailler l'aptitude du langage à remplir l'objectif prévu et toutes les mesures supplémentaires pour résoudre les inconvénients identifiés du langage.

7.4.3.3 Any changes required to the specified safety requirements of the E/E/PE safety-related system after applying 7.4.3.2 shall be agreed with the E/E/PE developer and documented.

NOTE – There will inevitably be iteration between the hardware and software architecture (see figure 5) and there is therefore a need to discuss with the hardware developer such issues as the test specification for the integration of the programmable electronics hardware and the software (see 7.5).

7.4.4 Requirements for support tools and programming languages

NOTE 1 – See also table A.3.

NOTE 2 – The selection of development tools will depend on the nature of the software development activities and the software architecture (see 7.4.3).

- For user application programming in a limited variability language, at low levels of safety integrity, the required tools and programming languages may be limited to the standard PLC languages, editors and loaders. Responsibility for compliance with 7.4.4 will therefore mainly rest with the supplier.
- At higher levels of safety integrity, restricted subsets of the PLC language may be necessary, and verification and validation tools such as code analysers and simulators needed. Responsibility will rest with both the supplier and the user in these circumstances.
- Tools for embedded applications using a full variability language will necessarily have to be more comprehensive even at low levels of safety integrity. Responsibility for conformance with 7.4.4 will rest here mainly with the software developer. This includes the PLC supplier who would use full variability languages in providing the low variability language for user application programming.

7.4.4.1 Depending on the nature of the software development, responsibility for conformance with 7.4.4 can rest with the supplier alone, or with the user alone, or with both (see note 2 above). The division of responsibility shall be documented during safety planning (see clause 6).

7.4.4.2 A suitable set of integrated tools, including languages, compilers, configuration management tools, and when applicable, automatic testing tools, shall be selected for the required safety integrity level. The availability of suitable tools (not necessarily those used during initial system development) to supply the relevant services over the whole lifetime of the E/E/PE safety-related system should be considered.

7.4.4.3 To the extent required by the safety integrity level, the programming language selected shall:

a) have a translator/compiler which has either a certificate of validation to a recognised national or international standard, or it shall be assessed to establish its fitness for purpose;

b) be completely and unambiguously defined or restricted to unambiguously defined features;

c) match the characteristics of the application;

d) contain features that facilitate the detection of programming mistakes;

e) support features that match the design method.

7.4.4.4 When 7.4.4.3 cannot be satisfied, then a justification for an alternative language used shall be documented during software architecture design description (see 7.4.3). The justification shall detail the fitness for purpose of the language, and any additional measures which address any identified shortcomings of the language.

7.4.4.5 Les règles de codage doivent être

a) revues par l'évaluateur pour vérifier qu'elles sont adéquates;

b) utilisées pour le développement de tous les logiciels relatifs à la sécurité.

7.4.4.6 Les règles de codage doivent spécifier une bonne pratique de la programmation, interdire les caractéristiques de langage peu sûres (par exemple les caractéristiques du langage non définies, les conceptions non structurées, etc.) et spécifier les procédures pour documenter le code source. Il convient que cette documentation contienne au moins les informations suivantes:

a) l'entité légale (exemple: la société, le ou les auteurs, etc.);

b) la description;

c) les entrées et sorties;

d) l'historique de la gestion de configuration.

7.4.5 Prescriptions concernant la conception détaillée et le développement

NOTE 1 – Voir aussi les tableaux A.4, B.1, B.7 et B.9.

NOTE 2 – La conception détaillée est définie ici comme étant la conception du système logiciel (partition des principaux composants de l'architecture en un système de modules logiciels), la conception des modules logiciels individuels et le codage. Dans les petites applications, la conception du système logiciel et celle de l'architecture peuvent être combinées.

NOTE 3 – La nature de la conception détaillée et du développement variera selon la nature des activités de développement du logiciel et l'architecture logicielle (voir 7.4.3). Pour la programmation des applications utilisateur à l'aide d'un langage de variabilité limitée, comme par exemple langage à contacts et blocs fonctionnels, la conception détaillée peut être considérée comme de la configuration plutôt que de la programmation. Cependant, il est aussi de bonne pratique de concevoir le logiciel de façon structurée, incluant l'organisation du logiciel en une structure modulaire séparant (autant que possible) les parties relatives à la sécurité; une vérification des limites et d'autres caractéristiques fournissant une protection contre les erreurs d'entrée de données; l'utilisation de modules déjà vérifiés et la fourniture d'une conception facilitant les modifications ultérieures du logiciel.

7.4.5.1 Suivant la nature du développement du logiciel, la responsabilité concernant le respect des prescriptions mentionnées en 7.4.5 peut incomber au fournisseur seul, à l'utilisateur seul, ou aux deux (voir note 3 ci-dessus). La répartition de la responsabilité doit être documentée pendant la planification de sécurité (voir article 6).

7.4.5.2 Il convient que les informations suivantes soient disponibles avant le début du processus de conception détaillée: la spécification des prescriptions de sécurité du logiciel (voir 7.2); la description de la conception de l'architecture logicielle (voir 7.4.3); le plan pour valider la sécurité du logiciel (voir 7.3).

7.4.5.3 Il convient que le logiciel soit produit en vue d'obtenir la modularité, la testabilité et la capacité de modification sûre.

7.4.5.4 Pour chaque composant/sous-système principal identifié dans la description de la conception de l'architecture logicielle (voir 7.4.3), un affinement supplémentaire de la conception doit être basé sur la partition en modules logiciels (c'est-à-dire la spécification de la conception du système logiciel). La conception de chaque module logiciel et les tests à appliquer à chaque module doivent être spécifiés.

NOTE – Pour les modules logiciels ou composants logiciels standard ou précédemment développés, aucune spécification de conception ou de test n'est nécessaire s'il peut être démontré que ces éléments satisfont aux prescriptions mentionnées en 7.4.2.11.

7.4.5.5 Il convient que les tests d'intégration du système logiciel appropriés soient spécifiés en vue de s'assurer que le système logiciel satisfait aux prescriptions de sécurité du logiciel spécifiées, correspondant au niveau d'intégrité de sécurité requis (voir 7.2).

7.4.4.5 Coding standards shall be

a) reviewed as fit for purpose by the assessor; and

b) used for the development of all safety-related software.

7.4.4.6 The coding standards shall specify good programming practice, proscribe unsafe language features (for example, undefined language features, unstructured designs, etc.) and specify procedures for source code documentation. As a minimum, the following information should be contained in the source code documentation:

a) legal entity (for example company, author(s), etc.);

b) description;

c) inputs and outputs;

d) configuration management history.

7.4.5 Requirements for detailed design and development

NOTE 1 – See also tables A.4, B.1, B.7 and B.9.

NOTE 2 – Detailed design is defined here to mean software system design – the partitioning of the major components in the architecture into a system of software modules, individual software module design, and coding. In small applications, software system design and architectural design may be combined.

NOTE 3 – The nature of detailed design and development will vary with the nature of the software development activities and the software architecture (see 7.4.3). For user application programming using a limited variability language, for example ladder logic and function blocks, detailed design can be considered as configuring rather than programming. However it is still good practice to design the software in a structured way, including organising the software into a modular structure that separates out (as far as possible) safety-related parts; including range checking and other features that provide protection against data input mistakes; using previously verified software modules; and providing a design that facilitates future software modifications.

7.4.5.1 Depending on the nature of the software development, responsibility for conformance with 7.4.5 can rest with the supplier alone, or with the user alone, or with both (see note 3 above). The division of responsibility shall be documented during safety planning (see clause 6).

7.4.5.2 The following information should be available prior to the start of detailed design: the specification of requirements for software safety (see 7.2); the description of the software architecture design (see 7.4.3); the plan for validating the software safety (see 7.3).

7.4.5.3 The software should be produced to achieve modularity, testability, and the capacity for safe modification.

7.4.5.4 For each major component/subsystem in the description of the software architecture design (see 7.4.3), further refinement of the design shall be based on a partitioning into software modules (i.e. the specification of the software system design). The design of each software module and the tests to be applied to each software module shall be specified.

NOTE – For standard or previously developed software components or software modules, no design or test specification is needed if it can be shown that they fulfil the requirements of 7.4.2.11.

7.4.5.5 Appropriate software system integration tests should be specified to ensure that the software system satisfies the specified requirements for software safety at the required safety integrity level (see 7.2).

7.4.6 Prescriptions concernant le codage

NOTE 1 – Voir aussi les tableaux A.4, B.1, B.7 et B.9.

7.4.6.1 Le code source doit

a) être lisible, compréhensible et testable;

b) satisfaire aux prescriptions spécifiées pour la conception des modules logiciels (voir 7.4.5);

c) satisfaire aux prescriptions spécifiées dans les règles de codage (voir 7.4.4);

d) satisfaire à toutes les prescriptions concernées identifiées pendant la planification de sécurité (voir article 6).

7.4.6.2 Il convient que chaque module de code logiciel fasse l'objet d'une revue.

NOTE – La revue de code est une activité de vérification (voir 7.9).

7.4.7 Prescriptions concernant le test des modules logiciels

NOTE – Le fait de s'assurer qu'un module logiciel satisfait à la spécification de test correspondante constitue une activité de vérification (voir 7.9). C'est la combinaison de la revue de code et du test de module logiciel qui fournit l'assurance qu'un module logiciel satisfait à la spécification correspondante, c'est-à-dire qu'il est vérifié.

7.4.7.1 Chaque module doit être testé comme spécifié pendant la conception du logiciel (voir 7.4.5).

7.4.7.2 Ces tests doivent montrer que chaque module logiciel remplit la fonction prévue et ne remplit aucune fonction non prévue.

NOTE 1 – Cela n'implique pas le test de toutes les combinaisons d'entrées, ni celui de toutes les combinaisons de sorties. Il est permis de faire le test de toutes les classes d'équivalence (voir C.5.7 de la CEI 61508-7) ou des tests basés sur la structure (voir C.5.8 de la CEI 61508-7). Il est admis que l'analyse des valeurs aux limites (voir C.5.4 de la CEI 61508-7), l'analyse de flot de contrôle (voir C.5.9 de la CEI 61508-7) ou l'analyse de chemin parasite (voir C.5.11 de la CEI 61508-7) réduisent les cas de test à un nombre acceptable. Les programmes analysables (voir C.2.7 de la CEI 61508-7) rendent les prescriptions plus faciles à satisfaire.

NOTE 2 – L'étendue de ces tests peut être réduite en cas de développement à l'aide de méthodes formelles (voir C.2.4 de la CEI 61508-7), de preuves formelles (voir C.5.13 de la CEI 61508-7) ou d'assertions (voir C.3.3 de la CEI 61508-7).

NOTE 3 – Il est permis d'utiliser aussi des techniques statistiques (voir annexe D de la CEI 61508-7).

7.4.7.3 Les résultats du test des modules logiciels doit être documenté.

7.4.7.4 Les procédures pour les actions correctives suite à l'échec d'un test doivent être spécifiées.

7.4.8 Prescriptions concernant l'intégration du logiciel

NOTE 1 – Voir également les tableaux A.5, B.2, B.3 et B.6.

NOTE 2 – Tester que le logiciel est correctement intégré constitue une activité de vérification (voir 7.9).

7.4.8.1 Les tests d'intégration du logiciel doivent être spécifiés durant la phase de conception et développement.

7.4.8.2 La spécification de test d'intégration du logiciel doit comprendre:

a) la répartition du logiciel en ensembles d'intégration gérables;

b) les cas de test et données de test;

c) les types de tests à effectuer;

d) l'environnement, les outils, la configuration et les programmes de test;

e) les critères de test sur lesquels on jugera la fin du test;

f) les procédures pour les actions correctives en cas d'échec d'un test.

7.4.6 Requirements for code implementation

NOTE – See also tables A.4, B.1, B.7 and B.9.

7.4.6.1 The source code shall

a) be readable, understandable and testable;

b) satisfy the specified requirements for software module design (see 7.4.5);

c) satisfy the specified requirements of the coding standards (see 7.4.4);

d) satisfy all relevant requirements specified during safety planning (see clause 6).

7.4.6.2 Each module of software code should be reviewed.

NOTE – Code review is a verification activity (see 7.9).

7.4.7 Requirements for software module testing

NOTE 1 – See also tables A.5, B.2, B.3 and B.6.

NOTE 2 – Testing that the software module correctly satisfies its test specification is a verification activity (see 7.9). It is the combination of code review and software module testing that provides assurance that a software module satisfies its associated specification, i.e. it is verified.

7.4.7.1 Each software module shall be tested as specified during software design (see 7.4.5).

7.4.7.2 These tests shall show that each software module performs its intended function and does not perform unintended functions.

NOTE 1 – This does not imply testing of all input combinations, nor of all output combinations. Testing all equivalence classes (see C.5.7 of IEC 61508-7) or structure based testing (see C.5.8 of IEC 61508-7) may be sufficient. Boundary value analysis (see C.5.4 of IEC 61508-7), control flow analysis (see C.5.9 of IEC 61508-7) or sneak circuit analysis (see C.5.11 of IEC 61508-7) may reduce the amount of test cases to an acceptable number. Analysable programs (see C.2.7 of IEC 61508-7) make the requirements easier to fulfil.

NOTE 2 – Where the development uses formal methods (see C.2.4 of IEC 61508-7), formal proofs (see C.5.13 of IEC 61508-7) or assertions (see C.3.3 of IEC 61508-7), such tests may be reduced in scope.

NOTE 3 – Statistical evidence may be used as well (see annex D of IEC 61508-7).

7.4.7.3 The results of the software module testing shall be documented.

7.4.7.4 The procedures for corrective action on failure of test shall be specified.

7.4.8 Requirements for software integration testing

NOTE 1 – See also tables A.5, B.2, B.3 and B.6.

NOTE 2 – Testing that the software is correctly integrated is a verification activity (see 7.9).

7.4.8.1 Software integration tests shall be specified concurrently during the design and development phase.

7.4.8.2 The specified software integration tests shall specify the following:

a) the division of the software into manageable integration sets;

b) test cases and test data;

c) types of tests to be performed;

d) test environment, tools, configuration and programs;

e) test criteria on which the completion of the test will be judged; and

f) procedures for corrective action on failure of test.

7.4.8.3 Le logiciel doit être testé conformément à la spécification de test d'intégration du logiciel. Ces tests doivent montrer que tous les modules logiciels et composants/sous-systèmes logiciels interagissent correctement de façon à réaliser leur fonction prévue et à ne réaliser aucune fonction non prévue.

NOTE 1 – Cela n'implique pas le test de toutes les combinaisons d'entrées, ni celui de toutes les combinaisons de sorties. Il est permis de faire le test de toutes les classes d'équivalence (voir C.5.7 de la CEI 61508-7) ou des tests basés sur la structure (voir C.5.8 de la CEI 61508-7). Il est admis que l'analyse des valeurs aux limites (voir C.5.4 de la CEI 61508-7), l'analyse de flot de contrôle (voir C.5.9 de la CEI 61508-7) ou l'analyse de chemin parasite (voir C.5.11 de la CEI 61508-7) réduisent les cas de test à un nombre acceptable. Les programmes analysables (voir C.2.7 de la CEI 61508-7) rendent les prescriptions plus faciles à satisfaire.

NOTE 2 – L'étendue de ces tests peut être réduite en cas de développement à l'aide de méthodes formelles (voir C.2.4 de la CEI 61508-7), de preuves formelles (voir C.5.13 de la CEI 61508-7) ou d'assertions (voir C.3.3 de la CEI 61508-7).

NOTE 3 – Il est permis d'utiliser aussi des techniques statistiques (voir annexe D de la CEI 61508-7).

7.4.8.4 Les résultats du test d'intégration du logiciel doivent être documentés en établissant les résultats du test et si les objectifs et les critères de test ont été atteints. En cas d'échec, les raisons de l'échec doivent être documentées.

7.4.8.5 Lors de l'intégration du logiciel, toute modification ou tout changement dans le logiciel doit faire l'objet d'une analyse d'impact qui doit déterminer tous les modules logiciels touchés et les activités nécessaires de reconception et revérification.

7.5 Intégration de l'électronique programmable (matériel et logiciel)

NOTE 1 – Voir aussi les tableaux A.6, B.3 et B.6.

NOTE 2 – Cette phase correspond à la case 9.4 de la figure 3.

7.5.1 Objectifs

7.5.1.1 Le premier objectif des prescriptions de ce paragraphe est d'intégrer le logiciel au matériel de l'électronique programmable cible.

7.5.1.2 Le deuxième objectif des prescriptions de ce paragraphe est de combiner le logiciel et le matériel de l'électronique programmable relative à la sécurité pour s'assurer de leur compatibilité et de leur conformité aux prescriptions du niveau d'intégrité de sécurité prévu.

NOTE 1 – Tester que le logiciel est correctement intégré au matériel de l'électronique programmable constitue une activité de vérification (voir 7.9).

NOTE 2 – Selon la nature de l'application, ces activités peuvent être combinées avec celles mentionnées en 7.4.8.

7.5.2 Prescriptions

7.5.2.1 Les tests d'intégration doivent être spécifiés pendant la phase de conception et développement afin de s'assurer de la compatibilité du matériel et du logiciel de l'électronique programmable relative à la sécurité.

NOTE – Une coopération étroite avec le développeur de l'E/E/PES peut être requise pour le développement des tests d'intégration.

7.5.2.2 Les tests d'intégration de l'électronique programmable (matériel et logiciel) doivent être spécifiés de façon à traiter les points suivants:

a) la division du système en niveaux d'intégration;
b) les cas de test et données de test;
c) les types de tests à effectuer;
d) la description de l'environnement de test comprenant les outils, le logiciel support et la configuration;
e) les critères suivant lesquels sera jugée la fin des tests.

7.4.8.3 The software shall be tested in accordance with the specified software integration tests. These tests shall show that all software modules and software components/subsystems interact correctly to perform their intended function and do not perform unintended functions.

NOTE 1 – This does not imply testing of all input combinations, nor of all output combinations. Testing all equivalence classes (see C.5.7 of IEC 61508-7) or structure based testing (see C.5.8 of IEC 61508-7) may be sufficient. Boundary value analysis (see C.5.4 of IEC 61508-7), control flow analysis (see C.5.9 of IEC 61508-7) or sneak circuit analysis (see C.5.11 of IEC 61508-7) may reduce the amount of test cases to an acceptable number. If development is carried out leading to analysable programs (see C.2.7 of IEC 61508-7), the requirements are easier to fulfil.

NOTE 2 – Where the development uses formal methods (see C.2.4 of IEC 61508-7), formal proofs (see C.5.13 of IEC 61508-7) or assertions (see C.3.3 of IEC 61508-7), such tests may be reduced in scope.

NOTE 3 – Statistical evidence may be used as well (see annex D of IEC 61508-7).

7.4.8.4 The results of software integration testing shall be documented, stating the test results, and whether the objectives and criteria of the test criteria have been met. If there is a failure, the reasons for the failure shall be documented.

7.4.8.5 During software integration, any modification or change to the software shall be subject to an impact analysis which shall determine all software modules impacted, and the necessary re-verification and re-design activities.

7.5 Programmable electronics integration (hardware and software)

NOTE 1 – See also tables A.6, B.3 and B.6.

NOTE 2 – This phase is box 9.4 of figure 3.

7.5.1 Objectives

7.5.1.1 The first objective of the requirements of this subclause is to integrate the software onto the target programmable electronic hardware.

7.5.1.2 The second objective of the requirements of this subclause is to combine the software and hardware in the safety-related programmable electronics to ensure their compatibility and to meet the requirements of the intended safety integrity level.

NOTE 1 – Testing that the software is correctly integrated with the programmable electronic hardware is a verification activity (see 7.9).

NOTE 2 – Depending on the nature of the application these activities may be combined with 7.4.8.

7.5.2 Requirements

7.5.2.1 Integration tests shall be specified during the design and development phase to ensure the compatibility of the hardware and software in the safety-related programmable electronics.

NOTE – Close co-operation with the developer of the E/E/PES may be required in order to develop the integration tests.

7.5.2.2 The integration tests for programmable electronics (hardware and software) shall specify the following:

a) the split of the system into integration levels;

b) test cases and test data;

c) types of tests to be performed;

d) test environment including tools, support software and configuration description;

e) test criteria on which the completion of the test will be judged.

7.5.2.3 Les tests d'intégration spécifiés pour l'électronique programmable (matériel et logiciel) doivent faire la distinction entre les activités que le développeur est capable d'effectuer sur son propre site et les activités nécessitant l'accès au site utilisateur.

7.5.2.4 Les tests d'intégration spécifiés pour l'électronique programmable (matériel et logiciel) doivent faire la distinction entre les activités suivantes:

a) l'intégration du système logiciel au matériel électronique programmable cible;

b) l'intégration E/E/PE, c'est-à-dire l'ajout d'interfaces telles que des capteurs et des actionneurs;

c) l'intégration totale de l'EUC et du système E/E/PE relatif à la sécurité.

NOTE – Les points b) et c) sont traités dans la CEI 61508-1 et la CEI 61508-2 et sont cités ici afin de placer l'élément (a) dans son contexte et fournir l'intégralité des informations.

7.5.2.5 Le logiciel doit être intégré au matériel de l'électronique programmable relative à la sécurité conformément aux tests d'intégration de l'électronique programmable (matériel et logiciel) spécifiés.

7.5.2.6 Lors du test d'intégration de l'électronique programmable relative à la sécurité (matériel et logiciel), toute modification ou changement apporté au système intégré doit être soumis à une analyse d'impact qui doit déterminer tous les modules logiciels touchés et les activités de revérification nécessaires.

7.5.2.7 Les cas de test et les résultats doivent être documentés pour analyse ultérieure.

7.5.2.8 Le test d'intégration de l'électronique programmable relative à la sécurité (matériel et logiciel) doit être documenté, en établissant les résultats du test et si les objectifs et les critères de test ont été atteints. En cas d'échec, les raisons de l'échec doivent être documentées. Toute modification ou tout changement résultant pour logiciel doit être soumis à une analyse d'impact qui doit identifier tous les composants/modules logiciels touchés et les activités de reconception et revalidation nécessaires.

7.6 Procédures d'exploitation et de modification du logiciel

NOTE 1 – Voir aussi tableau A.8.

NOTE 2 – Cette phase correspond à la case 9.5 de la figure 3.

7.6.1 Objectif

L'objectif des prescriptions du présent paragraphe est de fournir les informations et procédures concernant le logiciel nécessaire pour vérifier que la sécurité fonctionnelle du système E/E/PE relatif à la sécurité est maintenue pendant l'exploitation et les modifications.

7.6.2 Prescriptions

Les prescriptions sont données en 7.6 de la CEI 61508-2 et en 7.8 de la présente norme.

NOTE – Dans la présente norme, le logiciel (à la différence du matériel) ne peut être maintenu: il est toujours modifié.

7.7 Validation de sécurité du logiciel

NOTE 1 – Voir aussi les tableaux A.7, B.3 et B.5.

NOTE 2 – Cette phase correspond à la case 9.6 de la figure 3.

7.5.2.3 The specified integration tests for programmable electronics (hardware and software) shall distinguish between those activities which can be carried out by the developer on his premises and those that require access to the user's site.

7.5.2.4 The specified integration tests for programmable electronics (hardware and software) shall distinguish between the following activities:

a) merging of software system on to the target programmable electronic hardware;

b) E/E/PE integration, i.e. adding interfaces such as sensors and actuators;

c) total integration of the EUC and E/E/PE safety-related system.

NOTE – Items b) and c) are covered by IEC 61508-1 and IEC 61508-2 and are included here to put item a) in context and for completeness.

7.5.2.5 The software shall be integrated with the safety-related programmable electronic hardware in accordance with the specified integration tests for programmable electronics (hardware and software).

7.5.2.6 During the integration testing of the safety-related programmable electronics (hardware and software), any modification or change to the integrated system shall be subject to an impact analysis which shall determine all software modules impacted, and the necessary re-verification activities.

7.5.2.7 Test cases and their results shall be documented for subsequent analysis.

7.5.2.8 The integration testing of the safety-related programmable electronics (hardware and software) shall be documented, stating the test results, and whether the objectives and criteria of the test criteria have been met. If there is a failure, the reasons for the failure shall be documented. Any resulting modification or change to the software shall be subject to an impact analysis which shall determine all software components/modules impacted, and the necessary re-verification and re-design activities.

7.6 Software operation and modification procedures

NOTE 1 – See also table A.8.

NOTE 2 – This phase is box 9.5 of figure 3.

7.6.1 Objective

The objective of the requirements of this subclause is to provide information and procedures concerning software necessary to ensure that the functional safety of the E/E/PE safety-related system is maintained during operation and modification.

7.6.2 Requirements

The requirements are given in 7.6 of IEC 61508-2 and 7.8 of this standard.

NOTE – In this standard software (unlike hardware) is not capable of being maintained: it is always modified.

7.7 Software safety validation

NOTE 1 – See also tables A.7, B.3 and B.5.

NOTE 2 – This phase is box 9.6 of figure 3.

7.7.1 Objectif

L'objectif des prescriptions du présent paragraphe est de s'assurer que le système intégré est conforme aux prescriptions spécifiées pour la sécurité du logiciel (voir 7.2) au niveau d'intégrité de sécurité attendu.

7.7.2 Prescriptions

7.7.2.1 Si la conformité aux prescriptions pour la sécurité du logiciel a déjà été établie en tant que partie du système E/E/PE relatif à la sécurité (voir 7.7 de la CEI 61508-2), il n'est pas nécessaire de répéter la validation.

7.7.2.2 Les activités de validation doivent être effectuées comme spécifié pendant la planification de la validation de sécurité du logiciel (voir 7.3).

7.7.2.3 Les résultats de la validation de sécurité du logiciel doivent être documentés.

7.7.2.4 Pour chaque fonction de sécurité, la validation de sécurité du logiciel doit documenter les résultats suivants:

a) un enregistrement chronologique des activités de validation;

b) la version du plan de validation de sécurité du logiciel utilisée (voir 7.3);

c) la fonction de sécurité soumise à validation (par test ou analyse), avec la référence au plan de validation de sécurité du logiciel (voir 7.3);

d) les outils et les équipements utilisés avec les données d'étalonnage;

e) les résultats de l'activité de validation;

f) les différences entres les résultats prévus et les résultats obtenus.

7.7.2.5 Si les résultats obtenus ne correspondent pas aux résultats prévus, l'analyse effectuée ainsi que les décisions de poursuivre la validation, ou d'émettre une demande de modification et de revenir à une phase antérieure du cycle de vie du développement doivent être documentées en tant que résultats de la validation de sécurité du logiciel.

NOTE – Les prescriptions mentionnées de 7.7.2.2 à 7.7.2.5 sont fondées sur les prescriptions générales mentionnées en 7.14 de la CEI 61508-1.

7.7.2.6 La validation d'un logiciel relatif à la sécurité doit satisfaire aux prescriptions suivantes:

a) le test doit être la méthode principale de validation pour le logiciel; il est permis de compléter les activités de validation par des opérations d'animation et de modélisation;

b) le logiciel doit être exécuté en simulant

- les signaux d'entrée présents en exploitation normale;
- les événements prévus;
- les conditions non désirées nécessitant une action du système.

c) le fournisseur et/ou le développeur doit fournir au développeur du système l'accessibilité aux résultats documentés de la validation de sécurité du logiciel ainsi que toute documentation utile afin de lui permettre de satisfaire aux prescriptions de la CEI 61508-1 et de la CEI 61508-2.

7.7.2.7 Les prescriptions concernant la qualification des outils logiciels sont les suivantes:

a) tous les équipements utilisés pour la validation doivent être qualifiés conformément à une spécification traçable par rapport à une norme internationale ou nationale (si de telles normes sont disponibles) ou à une procédure dûment reconnue;

7.7.1 Objective

The objective of the requirements of this subclause is to ensure that the integrated system complies with the specified requirements for software safety (see 7.2) at the intended safety integrity level.

7.7.2 Requirements

7.7.2.1 If the compliance with the requirements for software safety has already been established as part for the E/E/PE safety-related system (see 7.7 of IEC 61508-2), then the validation need not be repeated.

7.7.2.2 The validation activities shall be carried out as specified during software safety validation planning (see 7.3).

7.7.2.3 The results of software safety validation shall be documented.

7.7.2.4 For each safety function, software safety validation shall document the following results:

a) a chronological record of the validation activities;

b) the version of the software safety validation plan (see 7.3) being used;

c) the safety function being validated (by test or analysis), together with reference to the software safety validation plan (see 7.3);

d) tools and equipment used together with calibration data;

e) the results of the validation activity;

f) discrepancies between expected and actual results.

7.7.2.5 When discrepancies occur between expected and actual results, the analysis made and the decisions taken on whether to continue the validation, or to issue a change request and return to an earlier part of the development lifecycle, shall be documented as part of the results of the software safety validation.

NOTE – The requirements of 7.7.2.2 to 7.7.2.5 are based on the general requirements of 7.14 of IEC 61508-1.

7.7.2.6 The validation of safety-related software shall meet the following requirements:

a) testing shall be the main validation method for software; animation and modelling may be used to supplement the validation activities;

b) the software shall be exercised by simulation of
   - input signals present during normal operation,
   - anticipated occurrences,
   - undesired conditions requiring system action;

d) the supplier and/or developer shall make available the documented results of the software safety validation and all pertinent documentation to the system developer to enable him to meet the requirements of IEC 61508-1 and IEC 61508-2 of this standard.

7.7.2.7 Software tool qualification requirements are as follows:

a) all equipment used for validation shall be qualified according to a specification traceable to an international standard (if available), or to a national standard (if available), or to a well recognized procedure;

b) les équipements utilisés pour la validation du logiciel doivent être qualifiés de manière appropriée et il doit être montré que tous les outils utilisés, matériels ou logiciels, sont adaptés à leur fonction.

NOTE – Dans la présente norme, la qualification est l'activité qui montre qu'une spécification particulière est remplie plutôt que les procédures de test de conformité génériques qui s'appliqueraient à une spécification quelconque.

7.7.2.8 Les prescriptions concernant les résultats de validation du logiciel sont les suivantes:

a) les tests doivent montrer que toutes les prescriptions spécifiées pour la sécurité du logiciel (voir 7.2) sont remplies et que le système logiciel ne remplit aucune fonction non prévue;

b) les cas de test et les résultats doivent être documentés pour une analyse ultérieure et une évaluation indépendante, comme requis par le niveau d'intégrité de sécurité (voir 8.2.12 de la CEI 61508-1);

c) les résultats documentés de la validation de sécurité du logiciel doivent établir soit la validation effective du logiciel soit les raisons de la non-validation.

7.8 Modification du logiciel

NOTE 1 – Voir aussi tableau A.8.

NOTE 2 – Cette phase correspond à la case 9.5 de la figure 3.

7.8.1 Objectif

L'objectif des prescriptions du présent paragraphe est d'apporter des corrections, des améliorations et adaptations au logiciel validé en s'assurant du maintien du niveau d'intégrité de sécurité du logiciel requis.

NOTE – Dans la présente norme, le logiciel (à la différence du matériel) ne peut être maintenu: il est toujours modifié.

7.8.2 Prescriptions

7.8.2.1 Avant de lancer toute modification du logiciel, on doit s'assurer que les procédures de modification du logiciel sont disponibles (voir 7.16 de la CEI 61508-1).

NOTE 1 – Les paragraphes 7.8.2.1 à 7.8.2.9 s'appliquent principalement aux changements arrivant pendant la phase d'exploitation du logiciel. Il est permis aussi de les appliquer pendant les phases d'intégration de l'électronique programmable et d'installation globale et mise en service (voir 7.13 de la CEI 61508-1).

NOTE 2 – La figure 9 de la CEI 61508-1 montre un exemple de modèle de procédure de modification.

7.8.2.2 Une modification ne doit être lancée que si la demande de modification du logiciel a été agréée selon les procédures spécifiées pendant la planification de sécurité (voir article 6), qui décrit en détail

a) les dangers susceptibles d'être rencontrés;

b) la modification proposée;

c) les raisons de la modification.

NOTE – La demande de modification peut, par exemple, être motivée par

- une sécurité fonctionnelle inférieure à celle spécifiée;
- une anomalie systématique mise en évidence par expérimentation;
- une législation de sécurité nouvelle ou modifiée;
- des modifications apportées à l'EUC ou à son utilisation;
- une modification des prescriptions globales de sécurité;
- une analyse des performances d'exploitation et de maintenance, indiquant que ces performances sont inférieures à celles prévues;
- des audits de sécurité fonctionnelle systématiques.

b) equipment used for software validation shall be qualified appropriately and any tools used, hardware or software, shall be shown to be suitable for purpose.

NOTE – In this standard, qualification is the activity which illustrates that a particular specification is met, rather than the generic conformance testing procedures which would apply to any specification.

7.7.2.8 Software validation result requirements are as follows:

a) the tests shall show that all of the specified requirements for software safety (see 7.2) are correctly performed and the software system does not perform unintended functions;

b) test cases and their results shall be documented for subsequent analysis and independent assessment as required by the safety integrity level (see 8.2.12 of IEC 61508-1);

c) the documented results of software safety validation shall state either that the software has passed the validation or the reasons for its failure.

7.8 Software modification

NOTE 1 – See also table A.8.

NOTE 2 – This phase is box 9.5 of figure 3.

7.8.1 Objective

The objective of the requirements of this subclause is to make corrections, enhancements or adaptations to the validated software, ensuring that the required software safety integrity level is sustained.

NOTE – In this standard software (unlike hardware) is not capable of being maintained; it is always modified.

7.8.2 Requirements

7.8.2.1 Prior to carrying out any software modification, software modification procedures shall be made available (see 7.16 of IEC 61508-1).

NOTE 1 – Subclauses 7.8.2.1 to 7.8.2.9 apply primarily to changes occurring during the operational phase of the software. They may also apply during the programmable electronics integration and overall installation and commissioning phases (see 7.13 of IEC 61508-1).

NOTE 2 – An example of a modification procedure model is shown in figure 9 of IEC 61508-1.

7.8.2.2 A modification shall be initiated only on the issue of an authorised software modification request under the procedures specified during safety planning (see clause 6) which details the following

a) the hazards which may be affected;

b) the proposed change;

c) the reasons for change.

NOTE – The reason for the request for the modification could arise from, for example

- functional safety below that specified;
- systematic fault experience;
- new or amended safety legislation;
- modifications to the EUC or its use;
- modification to the overall safety requirements;
- analysis of operations and maintenance performance, indicating that the performance is below target;
- routine functional safety audits.

7.8.2.3 Une analyse portant sur l'impact de la modification du logiciel proposée sur la sécurité fonctionnelle du système E/E/PE relatif à la sécurité doit être effectuée

a) pour déterminer si une analyse de danger et de risque est nécessaire ou non;

b) pour déterminer quelles phases du cycle de vie de sécurité du logiciel auront besoin d'être répétées.

7.8.2.4 Les résultats de l'analyse d'impact obtenus en 7.8.2.3 doivent être documentés.

7.8.2.5 Toutes les modifications qui ont un impact sur la sécurité fonctionnelle du système E/E/PE relatif à la sécurité doivent donner lieu à un retour à une phase adéquate du cycle de vie de sécurité du logiciel. Toutes les phases ultérieures doivent ensuite être exécutées conformément aux procédures spécifiées pour les phases spécifiques selon les prescriptions de la présente norme. Il convient que la planification de sécurité (voir article 6) détaille toutes les activités ultérieures.

NOTE – Il peut s'avérer nécessaire d'effectuer une analyse complète de danger et de risque, qui peut entraîner la nécessité de modifier les niveaux d'intégrité de sécurité par rapport à ceux spécifiés pour les systèmes relatifs à la sécurité et les dispositifs externes de réduction de risque.

7.8.2.6 La planification de sécurité pour la modification d'un logiciel relatif à la sécurité doit contenir les informations suivantes:

a) identification du personnel et spécification de leurs compétences requises;

b) spécification détaillée de la modification;

c) planification de la vérification;

d) domaine de revalidation et test de la modification dans les limites requises par le niveau d'intégrité de sécurité.

7.8.2.7 La modification doit être effectuée selon la planification.

7.8.2.8 Les détails de toute les modifications doivent être documentés en faisant référence à

a) la demande de modification/remise à niveau;

b) les résultats de l'analyse d'impact évaluant l'impact de la modification logicielle proposée sur la sécurité fonctionnelle ainsi que les décisions prises accompagnées de leurs justifications;

c) l'historique de la gestion de configuration logicielle;

d) les déviations par rapport à des modes d'exploitation et à des conditions normaux;

e) toutes les informations documentées affectées par l'activité de modification.

7.8.2.9 Les informations (par exemple, un journal) concernant le détail de toutes les modifications doivent être documentées. La documentation doit contenir la revérification et la revalidation des données ainsi que les résultats.

NOTE – Les paragraphes 7.8.2.1 à 7.8.2.9 s'appliquent principalement aux changements effectués pendant la phase d'exploitation du logiciel. Il est permis de les appliquer également pendant les phases d'intégration de l'électronique programmable et d'intégration globale et mise en service (voir 7.13 de la CEI 61508-1).

7.8.2.10 L'évaluation de l'activité de modification ou de rénovation requise doit dépendre des résultats de l'analyse d'impact et du niveau d'intégrité de sécurité du logiciel.

7.8.2.3 An analysis shall be carried out on the impact of the proposed software modification on the functional safety of the E/E/PE safety-related system

a) to determine whether or not a hazard and risk analysis is required;

b) to determine which software safety lifecycle phases will need to be repeated.

7.8.2.4 The impact analysis results obtained in 7.8.2.3 shall be documented.

7.8.2.5 All modifications which have an impact on the functional safety of the E/E/PE safety-related system shall initiate a return to an appropriate phase of the software safety lifecycle. All subsequent phases shall then be carried out in accordance with the procedures specified for the specific phases in accordance with the requirements in this standard. Safety planning (see clause 6) should detail all subsequent activities.

NOTE – It may be necessary to implement a full hazard and risk analysis, which may generate a need for different safety integrity levels than currently specified for the safety-related systems and external risk reduction facilities.

7.8.2.6 The safety planning for the modification of safety-related software shall include the following information:

a) identification of staff and specification of their required competency;

b) a detailed specification for the modification;

c) verification planning;

d) scope of revalidation and testing of the modification to the extent required by the safety integrity level.

7.8.2.7 Modification shall be carried out as planned.

7.8.2.8 Details of all modifications shall be documented, including references to

a) the modification/retrofit request;

b) the results of the impact analysis which assesses the impact of the proposed software modification on the functional safety, and the decisions taken with associated justifications;

c) software configuration management history;

d) deviation from normal operations and conditions;

e) all documented information affected by the modification activity.

7.8.2.9 Information (for example a log) on the details of all modifications shall be documented. The documentation shall include the reverification and revalidation of data and results.

NOTE – Subclauses 7.8.2.1 to 7.8.2.9 apply primarily to changes occurring during the operational phase of the software. They may also apply during the programmable electronics integration and overall installation and commissioning phases (see 7.13 of IEC 61508-1).

7.8.2.10 The assessment of the required modification or retrofit activity shall be dependent on the results of the impact analysis and the software safety integrity level.

7.9 Vérification du logiciel

NOTE – Voir aussi les tableaux A.9, B.2 et B.8.

7.9.1 Objectif

L'objectif des prescriptions du présent paragraphe est, dans les limites requises par le niveau d'intégrité de sécurité, de tester et d'évaluer les sorties d'une phase donnée du cycle de vie de sécurité du logiciel pour s'assurer de la conformité et la cohérence par rapport aux sorties et aux normes fournies en entrée de cette phase.

NOTE 1 – Le présent paragraphe prend en compte les aspects génériques de la vérification qui sont communs à plusieurs phases du cycle de vie de sécurité. Il n'implique aucune prescription supplémentaire pour les éléments de test des vérifications effectuées en 7.4.7 (test des modules logiciels), 7.4.8 (intégration du logiciel) et 7.5 (intégration de l'électronique programmable) et qui constituent en eux-mêmes des activités de vérification. De plus, aucune vérification n'est requise dans ce paragraphe, autre que la validation du logiciel (voir 7.7) qui, dans la présente norme, est le processus consistant à démontrer la conformité à la spécification des prescriptions de sécurité (vérification de bout en bout). Le contrôle de la spécification des prescriptions de sécurité proprement dite est effectué par des experts du domaine.

NOTE 2 – Selon l'architecture du logiciel, la responsabilité de l'activité de vérification peut être partagée entre toutes les organisations impliquées dans le développement et la modification du logiciel.

7.9.2 Prescriptions

7.9.2.1 La vérification du logiciel doit être planifiée (voir 7.4) parallèlement au développement, pour chaque phase du cycle de vie de sécurité du logiciel. Ces informations doivent être documentées.

7.9.2.2 La planification de la vérification du logiciel doit faire référence aux critères, techniques et outils à utiliser au cours des activités de vérification et doit comprendre

a) l'évaluation des prescriptions d'intégrité de sécurité;

b) la sélection et la documentation des stratégies, activités et techniques de vérification;

c) la sélection et l'utilisation des outils de vérification (banc de test, logiciel de test spécial, simulateurs d'entrées/sorties, etc.);

d) l'évaluation des résultats de la vérification;

e) les actions correctives à entreprendre.

7.9.2.3 La vérification du logiciel doit être effectuée selon la planification.

NOTE – La sélection des techniques, les mesures de vérification et le degré d'indépendance des activités de vérification dépendent d'un certain nombre de facteurs et peuvent être spécifiés dans les normes spécifiques de secteur d'application. Ces facteurs pourraient, par exemple, inclure

- la taille du projet,
- le degré de complexité,
- le degré de nouveauté de la conception,
- le degré de nouveauté technologique.

7.9.2.4 Les preuves permettant de montrer que la phase faisant l'objet d'une vérification a donné des résultats satisfaisants à tous égards doivent être documentées.

7.9.2.5 Après chaque vérification, il convient que la documentation de vérification comprenne

a) l'identification des éléments à vérifier;

b) l'identification des informations par rapport auxquelles la vérification a été effectuée;

c) les non-conformités.

NOTE – Des exemples de non-conformités comprennent les modules logiciels, les structures de données et les algorithmes peu adaptés au problème.

7.9 Software verification

NOTE – See also tables A.9, B.2 and B.8.

7.9.1 Objective

The objective of the requirements of this subclause is, to the extent required by the safety integrity level, to test and evaluate the outputs from a given software safety lifecycle phase to ensure correctness and consistency with respect to the outputs and standards provided as input to that phase.

NOTE 1 – This subclause considers the generic aspects of verification which are common to several safety lifecycle phases. This subclause does not place additional requirements for the testing element of verification in 7.4.7 (software module testing), 7.4.8 (software integration) and 7.5 (programmable electronics integration) which are verification activities in themselves. Nor does this subclause require verification in addition to software validation (see 7.7), which in this standard is the demonstration of conformance to the safety requirements specification (end-end verification). Checking whether the safety requirements specification is itself correct is carried out by domain experts.

NOTE 2 – Depending on the software architecture, responsibility for the verification activity may be split between all organisations involved in the development and modification of the software.

7.9.2 Requirements

7.9.2.1 The verification of software shall be planned (see 7.4) concurrently with the development, for each phase of the software safety lifecycle, and this information shall be documented.

7.9.2.2 The software verification planning shall refer to the criteria, techniques and tools to be used in the verification activities, and shall address

a) the evaluation of the safety integrity requirements;

b) the selection and documentation of verification strategies, activities and techniques;

c) the selection and utilisation of verification tools (test harness, special test software, input/output simulators etc.);

d) the evaluation of verification results;

e) the corrective actions to be taken.

7.9.2.3 The software verification shall be performed as planned.

NOTE – Selection of techniques, measures for verification and the degree of independence of the verification activities will depend upon a number of factors and may be specified in application sector standards. The factors could include, for example

- size of project;
- degree of complexity;
- degree of novelty of design;
- degree of novelty of technology.

7.9.2.4 Evidence shall be documented to show that the phase being verified has, in all respects, been satisfactorily completed.

7.9.2.5 After each verification, the verification documentation should include

a) identification of items to be verified;

b) identification of the information against which the verification has been done;

c) non-conformances.

NOTE – Examples of non-conformances include software modules, data structures, and algorithms poorly adapted to the problem.

7.9.2.6 Toutes les informations essentielles de la phase N du cycle de vie de sécurité du logiciel nécessaires à l'exécution correcte de la phase suivante N+1 doivent être disponibles et il convient de les vérifier. Les sorties de la phase N comprennent

a) l'adéquation de la spécification, de la description de la conception ou du code de la phase N en ce qui concerne
   - la fonctionnalité,
   - l'intégrité de sécurité, les performances et autres prescriptions de la planification de sécurité (voir article 6),
   - la lisibilité par l'équipe de développement,
   - la testabilité pour vérification ultérieure,
   - la modification sûre permettant une évolution ultérieure;
b) l'adéquation de la planification de la validation et/ou des tests spécifiés pour la phase N pour spécifier et décrire la conception de la phase N;
c) le contrôle des incompatibilités entre:
   - les tests spécifiés pendant la phase N et les tests spécifiés dans la phase précédente N–1,
   - les sorties de la phase N.

7.9.2.7 Sous la réserve des prescriptions mentionnées en 7.1.2.1, les activités de vérification suivantes doivent être effectuées:

a) vérification des prescriptions de sécurité du logiciel (voir 7.9.2.8);
b) vérification de l'architecture du logiciel (voir 7.9.2.9);
c) vérification de la conception du système logiciel (voir 7.9.2.10);
d) vérification de la conception des modules logiciels (voir 7.9.2.11);
e) vérification du code (voir 7.9.2.12);
f) vérification des données (voir 7.9.2.13);
g) test des modules logiciels (voir 7.4.7);
h) test d'intégration du logiciel (voir 7.4.8);
i) test d'intégration de l'électronique programmable (voir 7.5);
j) test des prescriptions de sécurité du logiciel (validation du logiciel) (voir 7.7).

7.9.2.8 Vérification des prescriptions de sécurité du logiciel: une fois que les prescriptions de sécurité du logiciel ont été spécifiées (voir 7.2), et avant que la phase suivante de conception et développement du logiciel soit lancée, la vérification doit

a) prendre en compte le fait que les prescriptions spécifiées de sécurité du logiciel (voir 7.2) satisfont de façon adéquate aux prescriptions de sécurité de l'E/E/PES spécifiées (voir CEI 61508-2) en ce qui concerne la fonctionnalité, l'intégrité de sécurité, les performances et les autres prescriptions de la planification de sécurité;
b) prendre en compte le fait que la planification de la validation de sécurité du logiciel (voir 7.3) satisfait de façon adéquate aux prescriptions de sécurité du logiciel spécifiées (voir 7.2);
c) contrôler les incompatibilités entre
   - les prescriptions de sécurité du logiciel spécifiées (voir 7.2) et les prescriptions de sécurité du système E/E/PES spécifiées (voir CEI 61508-2),
   - les prescriptions de sécurité du logiciel spécifiées (voir 7.2) et la planification de la validation de sécurité du logiciel (voir 7.3).

7.9.2.6 All essential information from phase N of the software safety lifecycle needed for the correct execution of the next phase N+1 shall be available and should be verified. Outputs from phase N include

a) adequacy of the specification, design description, or code in phase N for
   - functionality,
   - safety integrity, performance and other requirements of safety planning (see clause 6),
   - readability by the development team,
   - testability for further verification,
   - safe modification to permit further evolution;

b) adequacy of the validation planning and/or tests specified for phase N for specifying and describing the design of phase N;

c) check for incompatibilities between
   - the tests specified in phase N, and the tests specified in the previous phase N–1,
   - the outputs within phase N.

7.9.2.7 Subject to 7.1.2.1, the following verification activities shall be performed:

a) verification of software safety requirements (see 7.9.2.8);

b) verification of software architecture (see 7.9.2.9);

c) verification of software system design (see 7.9.2.10);

d) verification of software module design (see 7.9.2.11);

e) verification of code (see 7.9.2.12);

f) data verification (see 7.9.2.13);

g) software module testing (see 7.4.7);

h) software integration testing (see 7.4.8);

i) programmable electronics integration testing (see 7.5);

j) software safety requirements testing (software validation) (see 7.7).

7.9.2.8 Software safety requirements verification: once the software safety requirements have been specified (see 7.2), and before the next phase, software design and development begins, verification shall

a) consider whether the specified software safety requirements (see 7.2) adequately fulfil the specified E/E/PES safety requirements (see IEC 61508-2) for functionality, safety integrity, performance, and any other requirements of safety planning;

b) consider whether the software safety validation planning (see 7.3) adequately fulfils the specified software safety requirements (see 7.2);

c) check for incompatibilities between
   - the specified software safety requirements (see 7.2), and the specified E/E/PES safety requirements (see IEC 61508-2),
   - the specified software safety requirements (see 7.2), and the software safety validation planning (see 7.3).

7.9.2.9 Vérification de l'architecture du logiciel: après que la conception de l'architecture du logiciel a été établie, la vérification doit

a) prendre en compte le fait que la description de la conception de l'architecture du logiciel (voir 7.4.3) satisfait de façon adéquate aux prescriptions de sécurité du logiciel spécifiées (voir 7.2);

b) prendre en compte le fait que les tests d'intégration de l'architecture du logiciel spécifiées (voir 7.4.3) sont adaptés à la description de la conception de l'architecture du logiciel (voir 7.4.3);

c) prendre en compte le fait que les attributs de chaque composant/sous-système principal sont appropriés en matière de
   - faisabilité des performances de sécurité requises,
   - testabilité pour vérification ultérieure
   - lisibilité par l'équipe de développement et de vérification,
   - modification sûre permettant une évolution ultérieure;

d) contrôler les incompatibilités entre
   - la description de la conception de l'architecture du logiciel (voir 7.4.3) et les prescriptions de sécurité du logiciel spécifiées (voir 7.2),
   - la description de la conception de l'architecture du logiciel (voir 7.4.3) et les tests d'intégration de l'architecture du logiciel spécifiés (voir 7.4.3),
   - les tests d'intégration de l'architecture du logiciel spécifiés (voir 7.4.3) et la planification de la validation de sécurité du logiciel (voir 7.3).

7.9.2.10 Vérification de la conception du système logiciel: après que la conception du système logiciel a été spécifiée, la vérification doit

a) prendre en compte le fait que la conception spécifiée du système logiciel (voir 7.4.5) satisfait de façon adéquate à la conception de l'architecture du logiciel (voir 7.4.3);

b) prendre en compte le fait que les tests spécifiés de l'intégration du système logiciel (voir 7.4.5) satisfont de façon adéquate à la conception spécifiée du système logiciel (voir 7.4.5);

c) prendre en compte le fait que les attributs de chaque composant principal de la conception spécifiée du système logiciel (voir 7.4.5) sont appropriés en matière de
   - faisabilité des performances de sécurité requises,
   - testabilité pour vérification ultérieure,
   - lisibilité par l'équipe de développement et de vérification,
   - modification sûre permettant une évolution ultérieure;

   NOTE – Les tests d'intégration du système logiciel peuvent être spécifiés comme faisant partie des tests d'intégration de l'architecture du logiciel.

d) contrôler les incompatibilités entre
   - la conception spécifiée du système logiciel (voir 7.4.5) et la description de la conception de l'architecture du logiciel (voir 7.4.3),
   - la description de la conception du système logiciel (voir 7.4.5) et les tests spécifiés de l'intégration du système logiciel (voir 7.4.5),
   - les tests spécifiés de l'intégration du système logiciel (voir 7.4.5) et les tests spécifiés de l'intégration d'architecture (voir 7.4.3).

7.9.2.9 Software architecture verification: after the software architecture design has been established, verification shall

a) consider whether the description of the software architecture design (see 7.4.3) adequately fulfils the specified software safety requirements (see 7.2);

b) consider whether the specified tests of the software architecture integration (see 7.4.3) are adequate for the description of the software architecture design (see 7.4.3);

c) consider whether the attributes of each major component/subsystem is adequate with reference to

- feasibility of the safety performance required;
- testability for further verification;
- readability by the development and verification team;
- safe modification to permit further evolution;

d) check for incompatibilities between the following

- the description of the software architecture design (see 7.4.3), and the specified software safety requirements (see 7.2),
- the description of the software architecture design (see 7.4.3), and the specified tests of the software architecture integration (see 7.4.3),
- the specified tests of the software architecture integration (see 7.4.3), and the software safety validation planning (see 7.3).

7.9.2.10 Software system design verification: after the software system design has been specified, verification shall

a) consider whether the specified software system design (see 7.4.5) adequately fulfils the software architecture design (see 7.4.3);

b) consider whether the specified tests of the software system integration (see 7.4.5) adequately fulfil the specified software system design (see 7.4.5);

c) consider whether the attributes of each major component of the specified software system design (see 7.4.5) are adequate with reference to

- feasibility of the safety performance required;
- testability for further verification;
- readability by the development and verification team;
- safe modification to permit further evolution;

NOTE – The software system integration tests may be specified as part of the software architecture integration tests.

d) check for incompatibilities between

- the specified software system design (see 7.4.5), and the description of the software architecture design (see 7.4.3),
- the description of the software system design (see 7.4.5), and the specified tests of the software system integration (see 7.4.5),
- the specified tests of the software system integration (see 7.4.5) and the specified tests of the architecture integration (see 7.4.3).

7.9.2.11 Vérification de la conception des modules logiciels: lorsque la conception de chaque module logiciel a été spécifiée, la vérification doit

a) prendre en compte le fait que la conception spécifiée des modules logiciels (voir 7.4.5) satisfait de façon adéquate à la conception spécifiée du système logiciel (voir 7.4.5);

b) prendre en compte le fait que les tests spécifiés pour chaque module logiciel (voir 7.4.5) sont adaptés à la conception spécifiée des modules logiciels (voir 7.4.5);

c) prendre en compte le fait que les attributs de chaque module logiciel sont appropriés en matière de:

   – faisabilité des performances de sécurité requises (voir 7.2);
   – testabilité pour vérification ultérieure;
   – lisibilité par l'équipe de développement et de vérification;
   – modification sûre permettant une évolution ultérieure;

d) vérifier les incompatibilités entre

   – la conception spécifiée des modules logiciels (voir 7.4.5) et la conception spécifiée du système logiciel (voir 7.4.5);
   – (pour chaque module logiciel) la conception spécifiée des modules logiciels (voir 7.4.5) et les tests spécifiés des modules logiciels (voir 7.4.5);
   – les tests spécifiés des modules logiciels (voir 7.4.5) et les tests spécifiés de l'intégration du système logiciel (voir 7.4.5).

7.9.2.12 Vérification du code: le code source doit être vérifié par des méthodes statiques permettant de s'assurer de la conformité à la conception spécifiée du module logiciel (voir 7.4.5), aux règles de codage requises (voir 7.4.4) et aux prescriptions de la planification de sécurité (voir 7.3).

NOTE – Au cours des premières phases du cycle de vie de sécurité du logiciel, la vérification est statique (par exemple, inspection, revue, preuve formelle, etc.). La vérification du code comprend des techniques telles que les inspections du logiciel et les lectures croisées. C'est la combinaison des résultats de la vérification du code et du test des modules logiciels qui permet de s'assurer que chaque module logiciel satisfait à la spécification qui lui est associée. A partir de là, les tests deviennent le principal moyen de vérification.

7.9.2.13 Vérification des données

a) Les structures de données spécifiées au cours de la conception doivent être vérifiées en ce qui concerne

   – la complétude;
   – l'autocohérence;
   – la protection contre la détérioration ou l'altération;
   – la cohérence avec les prescriptions fonctionnelles du système orienté données.

b) Les données d'application doivent être vérifiées en ce qui concerne

   – la cohérence avec les structures de données;
   – la complétude;
   – la compatibilité avec le logiciel système de base (par exemple: séquence d'exécution, durée d'exécution, etc.);
   – l'exactitude des valeurs de données.

NOTE – En tant qu'exemple de données d'application, on peut citer le programme de pièces prévu pour une machine à commande numérique. Le logiciel système (typiquement un ensemble de sous-programmes) agit comme un interpréteur vis-à-vis des données d'application. Dans d'autres contextes, ces données d'application seraient considérées comme un programme d'application.

7.9.2.11 Software module design verification: after the design of each software module has been specified, verification shall

a) consider whether the specified software module design (see 7.4.5) adequately fulfils the specified software system design (see 7.4.5);

b) consider whether the specified tests for each software module (see 7.4.5) are adequate for the specified software module design (see 7.4.5);

c) consider whether the attributes of each software module are adequate with reference to
   - feasibility of the safety performance required (see 7.2);
   - testability for further verification;
   - readability by the development and verification team;
   - safe modification to permit further evolution;

d) check for incompatibilities between
   - the specified software module design (see 7.4.5), and the specified software system design (see 7.4.5),
   - (for each software module) the specified software module design (see 7.4.5), and the specified software module tests (see 7.4.5),
   - the specified software module tests (see 7.4.5), and the specified tests of the software system integration (see 7.4.5).

7.9.2.12 Code verification: the source code shall be verified by static methods to ensure conformance to the specified design of the software module (see 7.4.5), the required coding standards (see 7.4.4), and the requirements of safety planning (see 7.3).

NOTE – In the early phases of the software safety lifecycle, verification is static (for example inspection, review, formal proof etc). Code verification includes such techniques as software inspections and walk-throughs. It is the combination of the results of code verification and software module testing that provides assurance that each software module satisfies its associated specification. From then onwards testing becomes the primary means of verification.

7.9.2.13 Data verification

a) The data structures specified during design shall be verified for
   - completeness;
   - self-consistency;
   - protection against alteration or corruption;
   - consistency with the functional requirements of the data-driven system.

b) The application data shall be verified for
   - consistency with the data structures;
   - completeness;
   - compatibility with the underlying system software (for example sequence of execution, run-time, etc.); and
   - correctness of the data values.

   NOTE – An example of application data is a parts program for a numerically controlled machine. System software (typically a collection of subroutines) acts as an interpreter to the application data. In other contexts, this application data would be considered an application program.

c) Tous les paramètres modifiables doit être vérifiés en ce qui concerne la protection contre:
   - les valeurs initiales invalides ou non définies;
   - les valeurs erronées, incohérentes ou illogiques;
   - les modifications non autorisées;
   - l'altération des données.

d) Toutes les interfaces de l'installation et leur logiciel associé (c'est-à-dire capteurs et actionneurs, interface hors-ligne: voir 7.2.2.11) doivent être vérifiés en ce qui concerne
   - la détection des défaillances d'interface prévues;
   - la tolérance aux défaillances d'interface prévues.

e) Toutes les interfaces de communication et leur logiciel associé doivent être vérifiés pour s'assurer qu'ils ont un niveau adéquat de
   - détection des défaillances;
   - protection contre l'altération;
   - validation des données.

8 Evaluation de la sécurité fonctionnelle

8.1 L'objectif et les prescriptions de l'article 8 de la CEI 61508-1 s'appliquent à l'évaluation de logiciel relatif à la sécurité.

8.2 Sauf indications contraires dans les normes internationales liées à des secteurs d'application, le niveau minimal d'indépendance du personnel chargé de l'évaluation de la sécurité fonctionnelle doit être celui spécifié en 8.2.12 de la CEI 61508-1.

8.3 Les résultats des activités mentionnées dans le tableau A.10 peuvent être utilisés pour l'évaluation de la sécurité fonctionnelle.

NOTE – Sélectionner des techniques dans les annexes A et B n'est pas suffisant pour garantir que l'intégrité de sécurité requise sera atteinte (voir 7.1.2.6). Il convient que l'évaluateur considère aussi

- la cohérence et la complémentarité des méthodes, langages et outils choisis pour l'ensemble du cycle de développement;
- si les développeurs utilisent des méthodes, langages et outils qu'ils comprennent parfaitement;
- si les méthodes, langages et outils sont bien adaptés aux problèmes spécifiques rencontrés pendant le développement.

c) All modifiable parameters shall be verified for protection against
   - invalid or undefined initial values;
   - erroneous, inconsistent or unreasonable values;
   - unauthorised changes;
   - data corruption.

d) All plant interfaces and associated software (i.e. sensors and actuators, and off-line interfaces: see 7.2.2.11) shall be verified for
   - detection of anticipated interface failures;
   - tolerance to anticipated interface failures.

e) All communications interfaces and associated software shall be verified for an adequate level of
   - failure detection;
   - protection against corruption;
   - data validation.

8 Functional safety assessment

8.1 The objective and requirements of clause 8 of IEC 61508-1 apply to the assessment of safety-related software.

8.2 Unless otherwise stated in application sector international standards, the minimum level of independence of those carrying out the functional safety assessment shall be as specified in 8.2.12 of IEC 61508-1.

8.3 An assessment of functional safety may make use of the results of the activities of table A.10.

NOTE – Selecting techniques from annexes A and B does not guarantee by itself that the required safety integrity will be achieved (see 7.1.2.6). The assessor should also consider:

- the consistency and the complementary of the chosen methods, languages and tools for the whole development cycle;
- whether the developers use methods, languages and tools they fully understand;
- whether the methods, languages and tools are well-adapted to the specific problems encountered during development.

Annexe A
(normative)

Guide de sélection de techniques et mesures

Certains paragraphes de la présente norme sont associés à un tableau. Par exemple, le paragraphe 7.2 (spécification des prescriptions de sécurité du logiciel) est associé au tableau A.1. L'annexe B comporte des tableaux plus détaillés qui reprennent certaines entrées des tableaux de l'annexe A. Par exemple, le tableau B.2 reprend le thème du test et de l'analyse dynamique du tableau A.5.

Voir la CEI 61508-7 pour une présentation des techniques et mesures (au sens dispositions) spécifiques mentionnées dans les annexes A et B.

Chaque technique ou mesure mentionnée dans les tableaux est assortie d'une recommandation pour les niveaux d'intégrité de sécurité (SIL) 1 à 4. Ces recommandations se présentent de la manière suivante:

– HR: la technique ou mesure est hautement recommandée pour ce niveau d'intégrité de sécurité. Si cette technique ou mesure n'est pas utilisée, il convient que la raison de cette non-utilisation soit détaillée pendant la planification de sécurité et reçoive l'accord de l'évaluateur.

– R: la technique ou mesure est recommandée pour ce niveau d'intégrité de sécurité. Il s'agit d'une recommandation de niveau moins élevé qu'une recommandation HR.

– ---: l'utilisation de la technique ou mesure n'est ni recommandée ni déconseillée.

– NR: la technique ou mesure n'est absolument pas recommandée pour ce niveau d'intégrité de sécurité. Si cette technique ou mesure est utilisée, il convient que la raison de cette utilisation soit détaillée pendant la planification de sécurité et reçoive l'accord de l'évaluateur.

Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être satisfaite.

La classification des techniques et des mesures est liée au concept *d'efficacité* défini dans la CEI 61508-2. Lorsque tous les autres facteurs sont égaux, les techniques classées HR seront plus efficaces que celles classée R soit pour empêcher l'introduction d'anomalies systématiques lors du développement du logiciel soit (dans le cas de l'architecture du logiciel) pour maîtriser des anomalies résiduelles du logiciel révélées en cours d'exécution.

En raison du nombre élevé de facteurs pouvant affecter l'intégrité de sécurité du logiciel, il n'est pas possible d'élaborer un algorithme combinant les techniques et mesures qui seront correctes pour toute application donnée. Toutefois, la CEI 61508-6 guide l'utilisation des tableaux en donnant deux exemples de travail.

Pour une application particulière, la combinaison appropriée de techniques ou de mesures doit être établie pendant la planification de sécurité en sélectionnant les techniques ou mesures appropriées, sauf si d'autres prescriptions figurent dans les notes attenant au tableau.

La CEI 61508-6 fournit un premier guide sur l'interprétation des tableaux concernant la programmation des applications utilisateur.

Annex A
(normative)

Guide to the selection of techniques and measures

Some of the subclauses of this standard have an associated table, for example 7.2 (software safety requirements specification) is associated with table A.1. More detailed tables in annex B expand upon some of the entries in the tables of annex A, for example table B.2 expands on the topic of dynamic analysis and testing in table A.5.

See IEC 61508-7 for an overview of the specific techniques and measures referenced in annexes A and B.

With each technique or measure in the tables there is a recommendation for safety integrity levels 1 to 4. These recommendations are as follows.

- HR: the technique or measure is highly recommended for this safety integrity level. If this technique or measure is not used then the rationale behind not using it should be detailed during the safety planning and agreed with the assessor.
- R: the technique or measure is recommended for this safety integrity level as a lower recommendation to a HR recommendation.
- ---: the technique or measure has no recommendation for or against being used.
- NR: the technique or measure is positively not recommended for this safety integrity level. If this technique or measure is used then the rationale behind using it should be detailed during the safety planning and agreed with the assessor.

Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.

The ranking of the techniques and measures is linked to the concept of *effectiveness* used in IEC 61508-2. For all other factors being equal, techniques which are ranked HR will be more effective in either preventing the introduction of systematic faults during software development, or (for the case of the software architecture) more effective in controlling residual faults in the software revealed during execution than techniques ranked as R.

Given the large number of factors that affect software safety integrity it is not possible to give an algorithm for combining the techniques and measures that will be correct for any given application. However, guidance on the use of the tables by way of two worked examples is given in IEC 61508-6.

For a particular application, the appropriate combination of techniques or measures are to be stated during safety planning, with appropriate techniques or measures being selected unless the note attached to the table makes other requirements.

Initial guidance on the interpretation of the tables for user application programming is given in IEC 61508-6.

Tableau A.1 – Spécification des prescriptions de sécurité du logiciel (voir 7.2)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Outils de spécification assistée par ordinateur	B.2.4	R	R	HR	HR
2a	Méthodes semi-formelles	Tableau B.7	R	R	HR	HR
2b	Méthodes formelles comprenant, par exemple, CCS, CSP, HOL, LOTOS, OBJ, logique temporelle, VDM et Z	C.2.4	---	R	R	HR

NOTE 1 – La spécification des prescriptions de sécurité du logiciel nécessitera toujours une description du problème en langage naturel et l'emploi de toute notation mathématique nécessaire qui représente l'application.

NOTE 2 – Le tableau indique les prescriptions supplémentaires pour spécifier les prescriptions d'intégrité de sécurité du logiciel de manière claire et précise.

* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être remplie.

Tableau A.2 – Conception et développement du logiciel: conception de l'architecture du logiciel (voir 7.4.3)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Détection d'anomalie et diagnostic	C.3.1	----	R	HR	HR
2	Codes de détection et correction d'erreurs	C.3.2	R	R	R	HR
3a	Programmation par assertion des défaillances	C.3.3	R	R	R	HR
3b	Techniques à base de dispositif externe de sécurité	C.3.4	---	R	R	R
3c	Programmation diversifiée	C.3.5	R	R	R	HR
3d	Bloc de récupération	C.3.6	R	R	R	R
3e	Récupération arrière	C.3.7	R	R	R	R
3f	Récupération avant	C.3.8	R	R	R	R
3g	Mécanismes de récupération d'anomalie par relance	C.3.9	R	R	R	HR
3h	Mémorisation de cas d'exécution	C.3.10	---	R	R	HR
4	Dégradation «élégante»	C.3.11	R	R	HR	HR
5	Intelligence artificielle – correction d'anomalie	C.3.12	---	NR	NR	NR
6	Reconfiguration dynamique	C.3.13	---	NR	NR	NR
7a	Méthodes structurées comprenant, par exemple, JSD, MASCOT, SADT et Yourdon.	C.2.1	HR	HR	HR	HR
7b	Méthodes semi-formelles	Tableau B.7	R	R	HR	HR
7c	Méthodes formelles comprenant, par exemple, CCS, CSP, HOL, LOTOS, OBJ, logique temporelle, VDM et Z	C.2.4	---	R	R	HR
8	Outils de spécification assistée par ordinateur	B.2.4	R	R	HR	HR

NOTE – Il convient de prendre en considération les mesures de ce tableau concernant la tolérance aux anomalies (maîtrise des défaillances) en relation avec les prescriptions concernant l'architecture et la maîtrise des défaillances pour le matériel des équipements électroniques programmables de la CEI 61508-2.

* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être remplie.

Table A.1 – Software safety requirements specification (see 7.2)

Technique/Measure*		Ref.	SIL1	SIL2	SIL3	SIL4
1	Computer-aided specification tools	B.2.4	R	R	HR	HR
2a	Semi-formal methods	Table B.7	R	R	HR	HR
2b	Formal methods including for example, CCS, CSP, HOL, LOTOS, OBJ, temporal logic, VDM and Z	C.2.4	---	R	R	HR

NOTE 1 – The software safety requirements specification will always require a description of the problem in natural language and any necessary mathematical notation that reflects the application.

NOTE 2 – The table reflects additional requirements for specifying the software safety requirements clearly and precisely.

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.

Table A.2 – Software design and development: software architecture design (see 7.4.3)

Technique/Measure*		Ref	SIL1	SIL2	SIL3	SIL4
1	Fault detection and diagnosis	C.3.1	---	R	HR	HR
2	Error detecting and correcting codes	C.3.2	R	R	R	HR
3a	Failure assertion programming	C.3.3	R	R	R	HR
3b	Safety bag techniques	C.3.4	---	R	R	R
3c	Diverse programming	C.3.5	R	R	R	HR
3d	Recovery block	C.3.6	R	R	R	R
3e	Backward recovery	C.3.7	R	R	R	R
3f	Forward recovery	C.3.8	R	R	R	R
3g	Re-try fault recovery mechanisms	C.3.9	R	R	R	HR
3h	Memorising executed cases	C.3.10	---	R	R	HR
4	Graceful degradation	C.3.11	R	R	HR	HR
5	Artificial intelligence - fault correction	C.3.12	---	NR	NR	NR
6	Dynamic reconfiguration	C.3.13	---	NR	NR	NR
7a	Structured methods including for example, JSD, MASCOT, SADT and Yourdon.	C.2.1	HR	HR	HR	HR
7b	Semi-formal methods	Table B.7	R	R	HR	HR
7c	Formal methods including for example, CCS, CSP, HOL, LOTOS, OBJ, temporal logic, VDM and Z	C.2.4	---	R	R	HR
8	Computer-aided specification tools	B.2.4	R	R	HR	HR

NOTE – The measures in this table concerning fault tolerance (control of failures) should be considered with the requirements for architecture and control of failures for the hardware of the programmable electronics in IEC 61508-2.

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.

Tableau A.3 – Conception et développement du logiciel: outils supports et langage de programmation (voir 7.4.4)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Langage de programmation adéquat	C.4.6	HR	HR	HR	HR
2	Langage de programmation fortement typé	C.4.1	HR	HR	HR	HR
3	Sous-ensemble de langage	C.4.2	---	---	HR	HR
4a	Outils certifiés	C.4.3	R	HR	HR	HR
4b	Outils dans lesquels on a une confiance accrue résultant de l'utilisation	C.4.4	HR	HR	HR	HR
5a	Traducteur certifié	C.4.3	R	HR	HR	HR
5b	Traducteur: confiance accrue résultant de l'utilisation	C.4.4	HR	HR	HR	HR
6	Bibliothèque de modules logiciels et composants éprouvés/vérifiés	C.4.5	R	HR	HR	HR

* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être satisfaite.

Tableau A.4 – Conception et développement du logiciel: conception détaillée (voir 7.4.5 et 7.4.6)

(Comprend la conception du système logiciel, la conception des modules logiciels et le codage)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1a	Méthodes structurées comprenant, par exemple, JSD, MASCOT, SADT et Yourdon	C.2.1	HR	HR	HR	HR
1b	Méthodes semi-formelles	Tableau B.7	R	HR	HR	HR
1c	Méthodes formelles comprenant, par exemple, CCS, CSP, HOL, LOTOS, OBJ, logique temporelle, VDM et Z	C.2.4	---	R	R	HR
2	Outils de conception assistée par ordinateur	B.3.5	R	R	HR	HR
3	Programmation défensive	C.2.5	---	R	HR	HR
4	Approche modulaire	Tableau B.9	HR	HR	HR	HR
5	Règles de conception et de codage	Tableau B.1	R	HR	HR	HR
6	Programmation structurée	C.2.7	HR	HR	HR	HR
7	Utilisation de modules logiciels et composants éprouvés/ vérifiés (si disponibles)	C.2.10 C.4.5	R	HR	HR	HR

* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être remplie.

Table A.3 – Software design and development: support tools and programming language (see 7.4.4)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Suitable programming language	C.4.6	HR	HR	HR	HR
2	Strongly typed programming language	C.4.1	HR	HR	HR	HR
3	Language subset	C.4.2	---	---	HR	HR
4a	Certificated tools	C.4.3	R	HR	HR	HR
4b	Tools: increased confidence from use	C.4.4	HR	HR	HR	HR
5a	Certificated translator	C.4.3	R	HR	HR	HR
5b	Translator: increased confidence from use	C.4.4	HR	HR	HR	HR
6	Library of trusted/verified software modules and components	C.4.5	R	HR	HR	HR

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.

Table A.4 – Software design and development: detailed design (see 7.4.5 and 7.4.6)
(This includes software system design, software module design and coding)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1a	Structured methods including for example, JSD, MASCOT, SADT and Yourdon	C.2.1	HR	HR	HR	HR
1b	Semi-formal methods	Table B.7	R	HR	HR	HR
1c	Formal methods including for example, CCS, CSP, HOL, LOTOS, OBJ, temporal logic, VDM and Z	C.2.4	---	R	R	HR
2	Computer-aided design tools	B.3.5	R	R	HR	HR
3	Defensive programming	C.2.5	---	R	HR	HR
4	Modular approach	Table B.9	HR	HR	HR	HR
5	Design and coding standards	Table B.1	R	HR	HR	HR
6	Structured programming	C.2.7	HR	HR	HR	HR
7	Use of trusted/verified software modules and components (if available)	C.2.10 C.4.5	R	HR	HR	HR

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.

Tableau A.5 – Conception et développement du logiciel: test et intégration des modules logiciels (voir 7.4.7 et 7.4.8)

Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1 Test probabiliste	C.5.1	---	R	R	HR
2 Analyse dynamique et test	B.6.5 Tableau B.2	R	HR	HR	HR
3 Enregistrement et analyse de données	C.5.2	HR	HR	HR	HR
4 Tests fonctionnel et boîte noire	B.5.1 B.5.2 Tableau B.3	HR	HR	HR	HR
5 Modélisation du fonctionnement	C.5.20 Tableau B.6	R	R	HR	HR
6 Test d'interface	C.5.3	R	R	HR	HR

NOTE 1 – Le test et l'intégration des modules logiciels sont des activités de vérification (voir tableau A.9).

NOTE 2 – Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.

* Une technique/mesure repérée par un nombre doit être sélectionnée en fonction du niveau d'intégrité de sécurité.

Tableau A.6 – Intégration de l'électronique programmable (matériel et logiciel) (voir 7.5)

Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1 Test fonctionnel et boîte noire	B.5.1 B.5.2 Tableau B.3	HR	HR	HR	HR
2 Modélisation du fonctionnement	C.5.20 Tableau B.6	R	R	HR	HR

NOTE 1 – L'intégration de l'électronique programmable constitue une activité de vérification (voir tableau A.9).

NOTE 2 – Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.

* Une technique/mesure repérée par un nombre doit être sélectionnée en fonction du niveau d'intégrité de sécurité.

Tableau A.7 – Validation de sécurité du logiciel (voir 7.7)

Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1 Test probabiliste	C.5.1	---	R	R	HR
2 Simulation/modélisation	Tableau B.5	R	R	HR	HR
3 Tests fonctionnel et boîte noire	B.5.1 B.5.2 Tableau B.3	HR	HR	HR	HR

* Une technique/mesure repérée par un nombre doit être sélectionnée en fonction du niveau d'intégrité de sécurité.

NOTE – Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.

Table A.5 – Software design and development: software module testing and integration (see 7.4.7 and 7.4.8)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Probabilistic testing	C.5.1	---	R	R	HR
2	Dynamic analysis and testing	B.6.5 Table B.2	R	HR	HR	HR
3	Data recording and analysis	C.5.2	HR	HR	HR	HR
4	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	HR	HR	HR
5	Performance testing	C.5.20 Table B.6	R	R	HR	HR
6	Interface testing	C.5.3	R	R	HR	HR
NOTE 1 – Software module and integration testing are verification activities (see table A.9). NOTE 2 – Appropriate techniques/measures shall be selected according to the safety integrity level.						
* A numbered technique/measure shall be selected according to the safety integrity level.						

Table A.6 – Programmable electronics integration (hardware and software) (see 7.5)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	HR	HR	HR
2	Performance testing	C.5.20 Table B.6	R	R	HR	HR
NOTE 1 – Programmable electronics integration is a verification activity (see table A.9). NOTE 2 – Appropriate techniques/measures shall be selected according to the safety integrity level.						
* A numbered technique/measure shall be selected according to the safety integrity level.						

Table A.7 – Software safety validation (see 7.7)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Probabilistic testing	C.5.1	---	R	R	HR
2	Simulation/modelling	Table B.5	R	R	HR	HR
3	Functional and black-box testing	B.5.1 B.5.2 Table B.3	HR	HR	HR	HR
* A numbered technique/measure shall be selected according to the safety integrity level. NOTE – Appropriate techniques/measures shall be selected according to the safety integrity level.						

Tableau A.8 – Modification du logiciel (voir 7.8)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Analyse d'impact	C.5.23	HR	HR	HR	HR
2	Revérification d'un module logiciel modifié	C.5.23	HR	HR	HR	HR
3	Revérification des modules logiciels affectés	C.5.23	R	HR	HR	HR
4	Revalidation du système complet	C.5.23	---	R	HR	HR
5	Gestion de configuration logicielle	C.5.24	HR	HR	HR	HR
6	Enregistrement et analyse de données	C.5.2	HR	HR	HR	HR

* Une technique/mesure repérée par un nombre doit être sélectionnée en fonction du niveau d'intégrité de sécurité.

Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.

Tableau A.9 – Vérification du logiciel (voir 7.9)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Preuve formelle	C.5.13	---	R	R	HR
2	Test probabiliste	C.5.1	---	R	R	HR
3	Analyse statique	B.6.4 Tableau B.8	R	HR	HR	HR
4	Analyse dynamique et test	B.6.5 Tableau B.2	R	HR	HR	HR
5	Métriques de complexité du logiciel	C.5.14	R	R	R	R
Test des modules logiciels et intégration		Voir tableau A.5				
Test d'intégration de l'électronique programmable		Voir tableau A.6				
Test du système logiciel (validation)		Voir tableau A.7				

NOTE 1 – Pour des raisons pratiques, toutes les activités de vérification ont été rassemblées dans le présent tableau. Toutefois, cela n'implique aucune prescription supplémentaire pour les éléments de test dynamique de vérification du tableau A.5 et du tableau A.6 qui constituent en eux-mêmes des activités de vérification. De plus, aucun test de vérification n'est requis dans ce tableau, en plus de la validation du logiciel (tableau A.7) qui, dans la présente norme, est la démonstration de la conformité à la spécification des prescriptions de sécurité (vérification de bout en bout).

NOTE 2 – La vérification couvre la CEI 61508-1, la CEI 61508-2 et la CEI 61508-3. La première vérification du système relatif à la sécurité est donc réalisée par rapport aux spécifications du niveau précédent du système.

NOTE 3 – Au cours des premières phases du cycle de vie de sécurité du logiciel, la vérification est statique, par exemple inspection, revue, contrôle formel, etc. Dès que du code est produit, il est possible de réaliser un test dynamique. C'est la combinaison des deux types d'informations qui est exigée pour la vérification. Par exemple, la vérification du code d'un module logiciel par des moyens statiques comprend les techniques d'inspection du logiciel, de lecture croisée, d'analyse statique, de preuve formelle, etc. La vérification du code par des moyens dynamiques comprend le test fonctionnel, le test boîte blanche et le test statistique. C'est la combinaison des deux types de preuves qui fournit l'assurance que chaque module logiciel satisfait à sa spécification.

* Une technique/mesure repérée par un nombre doit être sélectionnée en fonction du niveau d'intégrité de sécurité.

Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.

Table A.8 – Modification (see 7.8)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Impact analysis	C.5.23	HR	HR	HR	HR
2	Reverify changed software module	C.5.23	HR	HR	HR	HR
3	Reverify affected software modules	C.5.23	R	HR	HR	HR
4	Revalidate complete system	C.5.23	---	R	HR	HR
5	Software configuration management	C.5.24	HR	HR	HR	HR
6	Data recording and analysis	C.5.2	HR	HR	HR	HR

* A numbered technique/measure shall be selected according to the safety integrity level.

Appropriate techniques/measures shall be selected according to the safety integrity level.

Table A.9 – Software verification (see 7.9)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Formal proof	C.5.13	---	R	R	HR
2	Probabilistic testing	C.5.1	---	R	R	HR
3	Static analysis	B.6.4 Table B.8	R	HR	HR	HR
4	Dynamic analysis and testing	B.6.5 Table B.2	R	HR	HR	HR
5	Software complexity metrics	C.5.14	R	R	R	R

Software module testing and integration	See table A.5
Programmable electronics integration testing	See table A.6
Software system testing (validation)	See table A.7

NOTE 1 – For convenience all verification activities have been drawn together under this table. However, this does not place additional requirements for the dynamic testing element of verification in table A.5 and table A.6 which are verification activities in themselves. Nor does this table require verification testing in addition to software validation (see table A.7), which in this standard is the demonstration of conformance to the safety requirements specification (end-end verification).

NOTE 2 – Verification crosses the boundaries of IEC 61508-1, IEC 61508-2 and IEC 61508-3. Therefore the first verification of the safety-related system is against the earlier system level specifications.

NOTE 3 – In the early phases of the software safety lifecycle verification is static, for example inspection, review, formal proof. When code is produced dynamic testing becomes possible. It is the combination of both types of information that is required for verification. For example code verification of a software module by static means includes such techniques as software inspections, walk-throughs, static analysis, formal proof. Code verification by dynamic means includes functional testing, white-box testing, statistical testing. It is the combination of both types of evidence that provides assurance that each software module satisfies its associated specification.

* A numbered technique/measure shall be selected according to the safety integrity level.

Appropriate techniques/measures shall be selected according to the safety integrity level.

Tableau A.10 – Evaluation de sécurité fonctionnelle (voir article 8)

	Evaluation/Technique*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Liste de contrôle	B.2.5	R	R	R	R
2	Tables de décision/de vérité	C.6.1	R	R	R	R
3	Métriques de complexité du logiciel	C.5.14	R	R	R	R
4	Analyse des défaillances	Tableau B.4	R	R	HR	HR
5	Analyse des défaillances de cause commune d'un logiciel diversifié (si du logiciel diversifié est effectivement utilisé)	C.6.3	---	R	HR	HR
6	Diagramme de blocs de fiabilité	C.6.5	R	R	R	R

* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être satisfaite.

Table A.10 – Functional safety assessment (see clause 8)

	Assessment/Technique*	Ref	SIL1	SIL2	SIL3	SIL4
1	Checklists	B.2.5	R	R	R	R
2	Decision/truth tables	C.6.1	R	R	R	R
3	Software complexity metrics	C.5.14	R	R	R	R
4	Failure analysis	Table B.4	R	R	HR	HR
5	Common cause failure analysis of diverse software (if diverse software is actually used)	C.6.3	---	R	HR	HR
6	Reliability block diagram	C.6.5	R	R	R	R
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Annexe B
(normative)

Tableaux détaillés

NOTE – Les références renvoient aux descriptions détaillées des techniques/mesures figurant dans la CEI 61508-7.

Tableau B.1 – Règles de conception et de codage
(référencées dans le tableau A.4)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Utilisation de règles de codage	C.2.6.2	HR	HR	HR	HR
2	Pas d'objets dynamiques	C.2.6.3	R	HR	HR	HR
3a	Pas de variables dynamiques	C.2.6.3	---	R	HR	HR
3b	Contrôle en ligne pendant la création de variables dynamiques	C.2.6.4	---	R	HR	HR
4	Utilisation limitée des interruptions	C.2.6.5	R	R	HR	HR
5	Utilisation limitée des pointeurs	C.2.6.6	---	R	HR	HR
6	Utilisation limitée de la récursion	C.2.6.7	---	R	HR	HR
7	Pas de branchements inconditionnels dans les programmes en langages de haut niveau	C.2.6.2	R	HR	HR	HR
NOTE – L'application des mesures 2 et 3a n'est pas nécessaire en cas d'utilisation d'un compilateur qui assure qu'un espace mémoire suffisant est affecté avant exécution à tous les objets et variables dynamiques, ou qui introduit des contrôles d'allocation correcte de mémoire en ligne au moment de l'exécution.						
* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être remplie.						

Tableau B.2 – Analyse dynamique et test
(référencés dans les tableaux A.5 et A.9)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Exécution de cas de test à partir de l'analyse des valeurs aux limites	C.5.4	R	HR	HR	HR
2	Exécution de cas de test à partir de l'estimation des erreurs	C.5.5	R	R	R	R
3	Exécution de cas de test à partir de l'implantation d'erreurs	C.5.6	---	R	R	R
4	Modélisation du fonctionnement	C.5.20	R	R	R	HR
5	Classes d'équivalence et test des partitions d'entrée	C.5.7	R	R	R	HR
6	Tests basés sur la structure	C.5.8	R	R	HR	HR
NOTE – L'analyse des cas de test se fait au niveau du sous-système et est basée sur la spécification et/ou la spécification et le code.						
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.						

Annex B
(normative)

Detailed tables

NOTE – The references indicate detailed descriptions of techniques/measures in IEC 61508-7.

Table B.1 – Design and coding standards
(referenced by table A.4)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Use of coding standard	C.2.6.2	HR	HR	HR	HR
2	No dynamic objects	C.2.6.3	R	HR	HR	HR
3a	No dynamic variables	C.2.6.3	---	R	HR	HR
3b	Online checking of the installation of dynamic variables	C.2.6.4	---	R	HR	HR
4	Limited use of interrupts	C.2.6.5	R	R	HR	HR
5	Limited use of pointers	C.2.6.6	---	R	HR	HR
6	Limited use of recursion	C.2.6.7	---	R	HR	HR
7	No unconditional jumps in programs in higher level languages	C.2.6.2	R	HR	HR	HR
NOTE – Measures 2 and 3a do not need to be applied if a compiler is used which ensures that sufficient memory for all dynamic variables and objects will be allocated before runtime, or which inserts runtime checks for the correct online allocation of memory.						
* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.						

Table B.2 – Dynamic analysis and testing
(referenced by tables A.5 and A.9)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Test case execution from boundary value analysis	C.5.4	R	HR	HR	HR
2	Test case execution from error guessing	C.5.5	R	R	R	R
3	Test case execution from error seeding	C.5.6	---	R	R	R
4	Performance modelling	C.5.20	R	R	R	HR
5	Equivalence classes and input partition testing	C.5.7	R	R	R	HR
6	Structure-based testing	C.5.8	R	R	HR	HR
NOTE – The analysis for the test cases is at the subsystem level and is based on the specification and/or the specification and the code.						
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Tableau B.3 – Tests fonctionnel et boîte noire
(référencés dans les tableaux A.5, A.6 et A.7)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Exécution de cas de test à partir de diagrammes cause/conséquence	B.6.6.2	---	---	R	R
2	Prototypage/Animation	C.5.17	---	---	R	R
3	Analyse des valeurs aux limites	C.5.4	R	HR	HR	HR
4	Classes d'équivalence et test des partitions d'entrée	C.5.7	R	HR	HR	HR
5	Simulation du procédé	C.5.18	R	R	R	R
NOTE 1 – L'analyse des cas de test est effectuée au niveau du système logiciel et est basée uniquement sur la spécification.						
NOTE 2 – La complétude de la simulation dépendra du niveau d'intégrité de sécurité, de la complexité et de l'application.						
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau de sécurité.						

Tableau B.4 – Analyse de défaillance
(référencée dans le tableau A.10)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1a	Diagrammes cause/conséquence	B.6.6.2	R	R	R	R
1b	Analyse par arbre d'événement	B.6.6.3	R	R	R	R
2	Analyse par arbre de panne	B.6.6.5	R	R	HR	HR
3	Analyse des modes de défaillance, de leurs effets et de leur criticité	B.6.6.4	R	R	HR	HR
4	Simulation de Monte-Carlo	C.6.6	R	R	R	R
NOTE 1 – Il convient d'effectuer une analyse de danger préliminaire afin de classer le logiciel par rapport au niveau d'intégrité de sécurité le plus approprié.						
* Les techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité. Des techniques/mesures équivalentes ou de remplacement sont indiquées à l'aide d'une lettre placée à la suite du numéro. Une seule des techniques/mesures équivalentes ou de remplacement doit être remplie.						

Tableau B.5 – Modélisation
(référencée dans le tableau A.7)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Diagramme de flux de données	C.2.2	R	R	R	R
2	Automate à états finis	B.2.3.2	---	R	HR	HR
3	Méthodes formelles	C.2.4	---	R	R	HR
4	Modélisation du fonctionnement	C.5.20	R	HR	HR	HR
5	Réseaux de Pétri temporels	B.2.3.3	---	R	HR	HR
6	Prototypage/animation	C.5.17	R	R	R	R
7	Diagrammes de structures	C.2.3	R	R	R	HR
NOTE 1 – Si une technique spécifique n'est pas mentionnée dans le tableau, elle ne doit pas être considérée comme exclue. Il convient de s'assurer qu'elle est conforme à la présente norme internationale.						
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.						

Table B.3 – Functional and black-box testing
(referenced by tables A.5, A.6 and A.7)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Test case execution from cause consequence diagrams	B.6.6.2	---	---	R	R
2	Prototyping/animation	C.5.17	---	---	R	R
3	Boundary value analysis	C.5.4	R	HR	HR	HR
4	Equivalence classes and input partition testing	C.5.7	R	HR	HR	HR
5	Process simulation	C.5.18	R	R	R	R
NOTE 1 – The analysis for the test cases is at the software system level and is based on the specification only.						
NOTE 2 – The completeness of the simulation will depend upon the safety integrity level, complexity and application.						
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Table B.4 – Failure analysis
(referenced by table A.10)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1a	Cause consequence diagrams	B.6.6.2	R	R	R	R
1b	Event tree analysis	B.6.6.3	R	R	R	R
2	Fault tree analysis	B.6.6.5	R	R	HR	HR
3	Failure modes, effects and criticality analysis	B.6.6.4	R	R	HR	HR
4	Monte-Carlo simulation	C.6.6	R	R	R	R
NOTE – Preliminary hazard analysis should have already taken place in order to categorise the software into the most appropriate safety integrity level.						
* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. Only one of the alternate or equivalent techniques/measures has to be satisfied.						

Table B.5 – Modelling
(referenced by table A.7)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Data flow diagrams	C.2.2	R	R	R	R
2	Finite state machines	B.2.3.2	---	R	HR	HR
3	Formal methods	C.2.4	---	R	R	HR
4	Performance modelling	C.5.20	R	HR	HR	HR
5	Time Petri nets	B.2.3.3	---	R	HR	HR
6	Prototyping/animation	C.5.17	R	R	R	R
7	Structure diagrams	C.2.3	R	R	R	HR
NOTE – If a specific technique is not listed in the table, it must not be assumed that it is excluded from consideration. It should conform to this standard.						
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Tableau B.6 – Modélisation du fonctionnement
(référencé dans les tableaux A.5 et A.6)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Tests d'avalanche/de stress	C.5.21	R	R	HR	HR
2	Temps de réponse et contraintes mémoire	C.5.22	HR	HR	HR	HR
3	Prescriptions relatives au fonctionnement	C.5.19	HR	HR	HR	HR
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.						

Tableau B.7 – Méthodes semi-formelles
(référencées dans les tableaux A.1, A.2 et A.4)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Diagrammes de blocs logiques/fonctionnels	voir la note ci-dessous	R	R	HR	HR
2	Diagrammes de séquence	voir la note ci-dessous	R	R	HR	HR
3	Diagrammes de flux de données	C.2.2	R	R	R	R
4	Automates finis/diagrammes de changement d'états	B.2.3.2	R	R	HR	HR
5	Réseaux de Pétri temporels	B.2.3.3	R	R	HR	HR
6	Tables de décision/de vérité	C.6.1	R	R	HR	HR
NOTE – Les diagrammes de blocs logiques/fonctionnels et les diagrammes de séquence sont décrits dans la CEI 61131-3.						
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.						

Tableau B.8 – Analyse statique
(référencée dans le tableau A.9)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Analyse des valeurs aux limites	C.5.4	R	R	HR	HR
2	Listes de contrôle	B.2.5	R	R	R	R
3	Analyse du flux de commande	C.5.9	R	HR	HR	HR
4	Analyse du flux de données	C.5.10	R	HR	HR	HR
5	Estimation des erreurs	C.5.5	R	R	R	R
6	Inspection selon Fagan	C.5.15	---	R	R	HR
7	Analyse de circuit parasite	C.5.11	---	---	R	R
8	Exécution symbolique	C.5.12	R	R	HR	HR
9	Lectures croisées/revues de conception	C.5.16	HR	HR	HR	HR
* Des techniques/mesures appropriées doivent être sélectionnées en fonction du niveau d'intégrité de sécurité.						

Table B.6 – Performance testing
(referenced by tables A.5 and A.6)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Avalanche/stress testing	C.5.21	R	R	HR	HR
2	Response timings and memory constraints	C.5.22	HR	HR	HR	HR
3	Performance requirements	C.5.19	HR	HR	HR	HR
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Table B.7 – Semi-formal methods
(referenced by tables A.1, A.2 and A.4)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Logic/function block diagrams	see note below	R	R	HR	HR
2	Sequence diagrams	see note below	R	R	HR	HR
3	Data flow diagrams	C.2.2	R	R	R	R
4	Finite state machines/state transition diagrams	B.2.3.2	R	R	HR	HR
5	Time Petri nets	B.2.3.3	R	R	HR	HR
6	Decision/truth tables	C.6.1	R	R	HR	HR
NOTE – Logic/function block diagrams and sequence diagrams are described in IEC 61131-3.						
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Table B.8 – Static analysis
(referenced by table A.9)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Boundary value analysis	C.5.4	R	R	HR	HR
2	Checklists	B.2.5	R	R	R	R
3	Control flow analysis	C.5.9	R	HR	HR	HR
4	Data flow analysis	C.5.10	R	HR	HR	HR
5	Error guessing	C.5.5	R	R	R	R
6	Fagan inspections	C.5.15	---	R	R	HR
7	Sneak circuit analysis	C.5.11	---	---	R	R
8	Symbolic execution	C.5.12	R	R	HR	HR
9	Walk-throughs/design reviews	C.5.16	HR	HR	HR	HR
* Appropriate techniques/measures shall be selected according to the safety integrity level.						

Tableau B.9 – Approche modulaire
(référencée dans le tableau A.4)

	Technique/Mesure*	Réf.	SIL1	SIL2	SIL3	SIL4
1	Limitation de la taille des modules logiciels	C.2.9	HR	HR	HR	HR
2	Masquage/encapsulation des informations	C.2.8	R	HR	HR	HR
3	Limitation du nombre de paramètres	C.2.9	R	R	R	R
4	Un point d'entrée/un point de sortie dans les sous-programmes et les fonctions	C.2.9	HR	HR	HR	HR
5	Interface totalement définie	C.2.9	HR	HR	HR	HR
NOTE – Pour obtenir des informations sur ces techniques, à l'exception de la technique de masquage/encapsulation des informations, voir C.2.9 de la CEI 61508-7.						
* Aucune de ces techniques n'est vraisemblablement suffisante à elle seule. Toutes les techniques appropriées doivent être prises en considération.						

Table B.9 – Modular approach
(referenced by table A.4)

	Technique/Measure*	Ref	SIL1	SIL2	SIL3	SIL4
1	Software module size limit	C.2.9	HR	HR	HR	HR
2	Information hiding/encapsulation	C.2.8	R	HR	HR	HR
3	Parameter number limit	C.2.9	R	R	R	R
4	One entry/one exit point in subroutines and functions	C.2.9	HR	HR	HR	HR
5	Fully defined interface	C.2.9	HR	HR	HR	HR
NOTE – For information on all these techniques except information hiding/encapsulation, see C.2.9 of IEC 61508-7.						
* No single technique is likely to be sufficient. All appropriate techniques shall be considered.						

Annexe C
(informative)

Bibliographie

CEI 61151:1992, *Instrumentation nucléaire – Amplificateurs et préamplificateurs utilisés avec des détecteurs de rayonnements ionisants – Méthodes d'essai*

ISO/CEI 12207:1995, *Technologies de l'information – Processus du cycle de vie du logiciel*

ANSI/ISA S84:1996, *Application of safety instrumented systems for the process industries*

Annex C
(informative)

Bibliography

IEC 61151:1992, *Nuclear instrumentation – Amplifiers and preamplifiers used with detectors of ionizing radiation – Test procedures*

ISO/IEC 12207:1995, *Information technology – Software life cycle processes*

ANSI/ISA S84:1996, *Application of safety instrumented systems for the process industries*

Standards Survey

The IEC would like to offer you the best quality standards possible. To make sure that we continue to meet your needs, your feedback is essential. Would you please take a minute to answer the questions overleaf and fax them to us at +41 22 919 03 00 or mail them to the address below. Thank you!

Customer Service Centre (CSC)
International Electrotechnical Commission
3, rue de Varembé
1211 Genève 20
Switzerland

or

Fax to: **IEC**/CSC at +41 22 919 03 00

Thank you for your contribution to the standards-making process.

A Prioritaire

Nicht frankieren
Ne pas affranchir

Non affrancare
No stamp required

RÉPONSE PAYÉE

SUISSE

Customer Service Centre (CSC)
International Electrotechnical Commission
3, rue de Varembé
1211 GENEVA 20
Switzerland

Q1 Please report on **ONE STANDARD** and **ONE STANDARD ONLY**. Enter the exact number of the standard: *(e.g. 60601-1-1)*

.....

Q2 Please tell us in what capacity(ies) you bought the standard *(tick all that apply)*. I am the/a:

- purchasing agent ❑
- librarian ❑
- researcher ❑
- design engineer ❑
- safety engineer ❑
- testing engineer ❑
- marketing specialist ❑

other.....

Q3 I work for/in/as a: *(tick all that apply)*

- manufacturing ❑
- consultant ❑
- government ❑
- test/certification facility ❑
- public utility ❑
- education ❑
- military ❑

other.....

Q4 This standard will be used for: *(tick all that apply)*

- general reference ❑
- product research ❑
- product design/development ❑
- specifications ❑
- tenders ❑
- quality assessment ❑
- certification ❑
- technical documentation ❑
- thesis ❑
- manufacturing ❑

other.....

Q5 This standard meets my needs: *(tick one)*

- not at all ❑
- nearly ❑
- fairly well ❑
- exactly ❑

Q6 If you ticked NOT AT ALL in Question 5 the reason is: *(tick all that apply)*

- standard is out of date ❑
- standard is incomplete ❑
- standard is too academic ❑
- standard is too superficial ❑
- title is misleading ❑
- I made the wrong choice ❑

other .....

Q7 Please assess the standard in the following categories, using the numbers:
(1) unacceptable,
(2) below average,
(3) average,
(4) above average,
(5) exceptional,
(6) not applicable

timeliness .....
quality of writing.....
technical contents.....
logic of arrangement of contents .....
tables, charts, graphs, figures.....
other .....

Q8 I read/use the: *(tick one)*

- French text only ❑
- English text only ❑
- both English and French texts ❑

Q9 Please share any comment on any aspect of the IEC that you would like us to know:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Enquête sur les normes

La CEI ambitionne de vous offrir les meilleures normes possibles. Pour nous assurer que nous continuons à répondre à votre attente, nous avons besoin de quelques renseignements de votre part. Nous vous demandons simplement de consacrer un instant pour répondre au questionnaire ci-après et de nous le retourner par fax au +41 22 919 03 00 ou par courrier à l'adresse ci-dessous. Merci !

Centre du Service Clientèle (CSC)

Commission Electrotechnique Internationale
3, rue de Varembé
1211 Genève 20
Suisse

ou

Télécopie: **CEI**/CSC +41 22 919 03 00

Nous vous remercions de la contribution que vous voudrez bien apporter ainsi à la Normalisation Internationale.

A Prioritaire

Nicht frankieren
Ne pas affranchir

Non affrancare
No stamp required

RÉPONSE PAYÉE

SUISSE

Centre du Service Clientèle (CSC)
Commission Electrotechnique Internationale
3, rue de Varembé
1211 GENÈVE 20
Suisse

Q1 Veuillez ne mentionner qu'**UNE SEULE NORME** et indiquer son numéro exact: *(ex. 60601-1-1)*

.....

Q2 En tant qu'acheteur de cette norme, quelle est votre fonction? *(cochez tout ce qui convient)*
Je suis le/un:

- agent d'un service d'achat ❑
- bibliothécaire ❑
- chercheur ❑
- ingénieur concepteur ❑
- ingénieur sécurité ❑
- ingénieur d'essais ❑
- spécialiste en marketing ❑

autre(s).....

Q3 Je travaille: *(cochez tout ce qui convient)*

- dans l'industrie ❑
- comme consultant ❑
- pour un gouvernement ❑
- pour un organisme d'essais/ certification ❑
- dans un service public ❑
- dans l'enseignement ❑
- comme militaire ❑

autre(s).....

Q4 Cette norme sera utilisée pour/comme *(cochez tout ce qui convient)*

- ouvrage de référence ❑
- une recherche de produit ❑
- une étude/développement de produit ❑
- des spécifications ❑
- des soumissions ❑
- une évaluation de la qualité ❑
- une certification ❑
- une documentation technique ❑
- une thèse ❑
- la fabrication ❑

autre(s).....

Q5 Cette norme répond-elle à vos besoins: *(une seule réponse)*

- pas du tout ❑
- à peu près ❑
- assez bien ❑
- parfaitement ❑

Q6 Si vous avez répondu PAS DU TOUT à Q5, c'est pour la/les raison(s) suivantes: *(cochez tout ce qui convient)*

- la norme a besoin d'être révisée ❑
- la norme est incomplète ❑
- la norme est trop théorique ❑
- la norme est trop superficielle ❑
- le titre est équivoque ❑
- je n'ai pas fait le bon choix ❑

autre(s) .....

Q7 Veuillez évaluer chacun des critères ci-dessous en utilisant les chiffres
(1) inacceptable,
(2) au-dessous de la moyenne,
(3) moyen,
(4) au-dessus de la moyenne,
(5) exceptionnel,
(6) sans objet

publication en temps opportun .....
qualité de la rédaction.....
contenu technique .....
disposition logique du contenu .....
tableaux, diagrammes, graphiques, figures .....
autre(s) .....

Q8 Je lis/utilise: *(une seule réponse)*

- uniquement le texte français ❑
- uniquement le texte anglais ❑
- les textes anglais et français ❑

Q9 Veuillez nous faire part de vos observations éventuelles sur la CEI:

.....
.....
.....
.....
.....
.....

ISBN 2-8318-4618-8

ICS 25.040.40

Typeset and printed by the IEC Central Office
GENEVA, SWITZERLAND