NORME INTERNATIONALE

INTERNATIONAL STANDARD

CEI
IEC
60880-2

Première édition
First edition
2000-12

Logiciel pour les calculateurs de sûreté des centrales nucléaires –

Partie 2:
Défense contre les défaillances de cause commune provoquées par le logiciel, utilisation d'outils logiciels et de logiciels prédéveloppés

Software for computers important to safety for nuclear power plants –

Part 2:
Software aspects of defence against common cause failures, use of software tools and of pre-developed software

Numéro de référence
Reference number
CEI/IEC 60880-2:2000

Numérotation des publications

Depuis le 1er janvier 1997, les publications de la CEI sont numérotées à partir de 60000. Ainsi, la CEI 34-1 devient la CEI 60034-1.

Editions 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.

Informations supplémentaires sur les publications de la CEI

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 cette publication, y compris sa validité, sont disponibles dans le Catalogue des publications de la CEI (voir ci-dessous) en plus des nouvelles éditions, amendements et corrigenda. Des informations sur les sujets à l'étude et l'avancement des travaux entrepris par le comité d'études qui a élaboré cette publication, ainsi que la liste des publications parues, sont également disponibles par l'intermédiaire de:

- **Site web de la CEI (www.iec.ch)**
- **Catalogue des publications de la CEI**

  Le catalogue en ligne sur le site web de la CEI (www.iec.ch/catlg-f.htm) vous permet de faire des recherches en utilisant de nombreux critères, comprenant des recherches textuelles, par comité d'études ou date de publication. Des informations en ligne sont également disponibles sur les nouvelles publications, les publications remplacées ou retirées, ainsi que sur les corrigenda.
- **IEC Just Published**

  Ce résumé des dernières publications parues (www.iec.ch/JP.htm) est aussi disponible par courrier électronique. Veuillez prendre contact avec le Service client (voir ci-dessous) pour plus d'informations.
- **Service clients**

  Si vous avez des questions au sujet de cette publication ou avez besoin de renseignements supplémentaires, prenez contact avec le Service clients:

  Email: custserv@iec.ch
  Tél: +41 22 919 02 11
  Fax: +41 22 919 03 00

Publication numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series. For example, IEC 34-1 is now referred to as IEC 60034-1.

Consolidated editions

The IEC is now publishing consolidated versions of its publications. 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.

Further information on IEC publications

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 this publication, including its validity, is available in the IEC Catalogue of publications (see below) in addition to new editions, amendments and corrigenda. 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 also available from the following:

- **IEC Web Site (www.iec.ch)**
- **Catalogue of IEC publications**

  The on-line catalogue on the IEC web site (www.iec.ch/catlg-e.htm) enables you to search by a variety of criteria including text searches, technical committees and date of publication. On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.
- **IEC Just Published**

  This summary of recently issued publications (www.iec.ch/JP.htm) is also available by email. Please contact the Customer Service Centre (see below) for further information.
- **Customer Service Centre**

  If you have any questions regarding this publication or need further assistance, please contact the Customer Service Centre:

  Email: custserv@iec.ch
  Tel: +41 22 919 02 11
  Fax: +41 22 919 03 00

.

NORME INTERNATIONALE

INTERNATIONAL STANDARD

CEI
IEC
60880-2

Première édition
First edition
2000-12

Logiciel pour les calculateurs de sûreté des centrales nucléaires –

Partie 2: Défense contre les défaillances de cause commune provoquées par le logiciel, utilisation d'outils logiciels et de logiciels prédéveloppés

Software for computers important to safety for nuclear power plants –

Part 2: Software aspects of defence against common cause failures, use of software tools and of pre-developed software

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 W

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

SOMMAIRE

Pages

CONTENTS

Pages

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

LOGICIEL POUR LES CALCULATEURS DE SÛRETÉ DES CENTRALES NUCLÉAIRES –

Partie 2: Défense contre les défaillances de cause commune provoquées par le logiciel, utilisation d'outils logiciels et de logiciels prédéveloppés

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, spécifications techniques, 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 60880-2 a été établie par le sous-comité 45A: Instrumentation des réacteurs, du comité d'études 45 de la CEI: Instrumentation nucléaire.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
45A/402/FDIS	45A/406/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.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 3.

Les annexes A, B, C et D sont données uniquement à titre d'information.

Une fois révisée, la CEI 60880 (1986) paraîtra sous le numéro CEI 60880-1.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant 2006. A cette date, la publication sera

- reconduite;
- supprimée;
- remplacée par une édition révisée, ou
- amendée.

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

SOFTWARE FOR COMPUTERS IMPORTANT TO SAFETY FOR NUCLEAR POWER PLANTS –

Part 2: Software aspects of defence against common cause failures, use of software tools and of pre-developed software

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 specifications, 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 60880-2 has been prepared by subcommittee 45A: Reactor instrumentation, of IEC technical committee 45: Nuclear instrumentation.

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

FDIS	Report on voting
45A/402/FDIS	45A/406/RVD

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

This publication has been drafted in accordance with the ISO/IEC Directives, Part 3.

Annexes A, B, C and D are for information only.

When IEC 60880 (1986) is revised, it will be published as IEC 60880-1.

The committee has decided that the contents of this publication will remain unchanged until 2006. At this date, the publication will be

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

INTRODUCTION

La présente partie de la CEI 60880 énonce les prescriptions applicables aux logiciels pour systèmes informatiques de sûreté dans les centrales nucléaires. Elle doit être lue conjointement avec la CEI 60880. Cette partie comprend des exigences sur différents sujets; ces exigences ont leur origine dans les progrès technologiques et l'expérience acquise sur les systèmes logiciels jouant un rôle en matière de sûreté, depuis la parution de la CEI 60880.

Il faut mettre en oeuvre une défense contre les défauts logiciels pouvant provoquer une défaillance de cause commune (CCF) des systèmes informatiques conformément à la Publication de la Série Sécurité N° 50-C-D de l'AIEA, et les prescriptions techniques qui en découlent sont données. Les outils de support logiciel ont connu une croissance rapide et des recommandations liées à leur utilisation sont fournies. L'utilisation pratique de logiciels pour systèmes informatiques importants pour la sûreté repose souvent sur l'utilisation de logiciels prédéveloppés, et des prescriptions et des recommandations sont fournies dans ce sens.

La présente partie de la CEI 60880 fait référence à la publication fondamentale de sécurité, la CEI 61513.

Cette partie s'applique aux fonctions I&C de catégorie A et aux systèmes et équipements associés (FSE) de la CEI 61226.

NOTE Conformément au schéma de classification de la CEI 61513, les fonctions de catégorie A sont réalisées au sein de systèmes de classe 1, qui correspondent aux systèmes de sûreté de l'AIEA.

INTRODUCTION

This part of IEC 60880 provides requirements for software for computer-based safety systems in nuclear power plants and should be read in conjunction with IEC 60880. This part includes requirements on several topics arising from advances in technology and experience of software systems with a role in safety since the publication of IEC 60880.

Defence against software faults which can lead to Common Cause Failure (CCF) of computer-based systems has to be provided in accordance with the IAEA Safety Series No. 50-C-D, and technical requirements arising from these provisions are given. There has also been a rapid growth in the use of computerized support tools, and recommendations for their use are given. The practical use of software for computer-based safety systems often depends on the use of pre-developed software, and requirements and recommendations in this connection are given.

This part of IEC 60880 references the basic safety publication IEC 61513.

This part applies to Category A I&C functions and associated systems and equipment (FSE) of IEC 61226.

NOTE According to the classification scheme of IEC 61513, category A functions are implemented in class 1 systems, which correspond to the safety systems of the IAEA.

LOGICIEL POUR LES CALCULATEURS DE SÛRETÉ DES CENTRALES NUCLÉAIRES –

Partie 2: Défense contre les défaillances de cause commune provoquées par le logiciel, utilisation d'outils logiciels et de logiciels prédéveloppés

1 Domaine d'application et objet

Les prescriptions de la présente partie de la CEI 60880 sont applicables aux logiciels des systèmes informatiques de sûreté dans les centrales nucléaires. Elle fournit des prescriptions pour les logiciels des fonctions de catégorie A et des systèmes et équipements associés (FSE). Elle porte sur

- les moyens de défense contre les défaillances de mode commun provoquées par les logiciels;
- les outils automatisés pour le développement de logiciels importants pour la sûreté; et
- l'utilisation de logiciels préexistants.

La préparation et la validation des données sont traitées dans le cadre de ces trois thèmes.

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 60880. Pour les références datées, les amendements ultérieurs ou les révisions de ces publications ne s'appliquent pas. Toutefois, les parties prenantes aux accords fondés sur la présente partie de la CEI 60880 sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des documents normatifs indiqués ci-après. Pour les références non datées, la dernière édition du document normatif en référence s'applique. Les membres de l'ISO et de la CEI possèdent le registre des Normes internationales en vigueur.

CEI 60880:1986, *Logiciel pour les calculateurs utilisés dans les systèmes de sûreté des centrales nucléaires*

CEI 61226:1993, *Centrales nucléaires – Systèmes d'instrumentation et de contrôle-commande importants pour la sûreté – Classification*

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

CEI 61513:—, *Centrales nucléaires de puissance – Contrôle-commande des systèmes important pour la sûreté – Prescriptions générales pour les systèmes* [1)]

ISO/IEC 9126:1991, *Technologies de l'information – Evaluation des produits logiciels – Caractéristiques de qualité et directives d'utilisation*

AIEA 50-C-D (rev 1):1989, *Code pour la sûreté des centrales nucléaires: Conception*

AIEA 50-SG-D11:1986, *Principes généraux de sûreté dans la conception des centrales nucléaires: Guide de sûreté*

IEEE 610:1990, *Lexique de la terminologie d'ingénierie logicielle* (disponible en anglais seulement)

———————

[1)] A publier.

SOFTWARE FOR COMPUTERS IMPORTANT TO SAFETY FOR NUCLEAR POWER PLANTS –

Part 2: Software aspects of defence against common cause failures, use of software tools and of pre-developed software

1 Scope and object

This part of IEC 60880 is applicable to the software of computer-based safety systems in nuclear power plants. It gives requirements for software for category A functions, systems and associated equipment (FSE). It addresses

- defence against common cause failures, caused by software;
- automated tools for the development of software important to safety; and
- use of pre-developed software.

Preparation and confirmation of data is dealt with within these three topics as appropriate.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of IEC 60880. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of IEC 60880 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of IEC and ISO maintain registers of currently valid International Standards.

IEC 60880:1986, *Software for computers in the safety systems of nuclear power stations*

IEC 61226:1993, *Nuclear power plants – Instrumentation and control systems important for safety – Classification*

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

IEC 61513: —, *Nuclear power plants – Instrumentation and control for systems important to safety – General requirements for systems* [1)]

ISO/IEC 9126:1991, *Information technology – Software product evaluation – Quality characteristics and guidelines for their use*

IAEA 50-C-D (rev 1):1988, *Code on the safety of Nuclear Power Plants: Design*

IAEA 50-SG-D11:1986, *General design safety principles for nuclear power plants – A safety guide*

IEEE 610:1990, *Standard glossary of software engineering terminology*

---

1) To be published

3 Définitions et abréviations

Pour les besoins de la présente partie de la CEI 60880, les termes et définitions donnés dans la CEI 60880 et dans les publications de la Série Sécurité N° 50-C-D de l'AIEA, ainsi que les suivants, sont applicables.

NOTE 1 Les termes «doit/doivent», «il convient de» et «peut/peuvent» sont utilisés conformément aux prescriptions de la CEI.

NOTE 2 L'exemple suivant est donné afin de lever toute ambiguïté quant à l'utilisation des termes **erreur**, **défaut**, **défaillance** et **trajectoire de signal**:

Si une personne ou un processus commet une **erreur** dans la production de quelque chose, cela entraîne un **défaut** du produit. Lorsque le produit est utilisé, il peut être satisfaisant ou bien tomber en panne si le **défaut** n'est pas corrigé. Si l'utilisation sollicite le **défaut**, le produit tombera en panne si aucun autre moyen de défense n'empêche la **défaillance**. Une **défaillance** est due à la fois à un **défaut** et à une sollicitation, sans aucun autre moyen de défense en service. Dans le cas des logiciels, un **défaut** est sollicité par une **trajectoire de signal**.

3.1
animation
processus par lequel le comportement défini par une spécification est visualisé avec ses valeurs effectives dérivées des expressions de comportement établies et de certaines valeurs d'entrée

3.2
fonction d'application
fonction d'un système d'instrumentation et commande qui effectue une tâche relative au processus à commander plutôt qu'au fonctionnement du système lui-même
[dérivée de 2.1 de la CEI 60880]

3.3
canal
chemin séparé sur lequel les informations sont acheminées dans un système redondant ou réparti; ce chemin peut également avoir une redondance

3.4
défaillance de cause commune (CCF)
défaillance qui résulte d'un ou plusieurs événements, qui provoquent des défaillances simultanées de deux ou plusieurs canaux dans un système à canaux multiples ou dans plusieurs systèmes, conduisant à une défaillance du (des) système(s)
[3.6.10 de la CEI 61508-4, modifiée]

NOTE 1 Selon le contexte, une CCF peut être vue au niveau système ou au niveau des systèmes qui constituent un groupe de sûreté.

NOTE 2 Voir en 3.8 la définition de défaillance.

3.5
donnée
représentation d'une information ou d'instructions d'une manière adaptée pour la communication, l'interprétation ou le traitement au moyen d'un calculateur
[définition adaptée de IEEE 610]

NOTE Les données qui sont nécessaires pour définir des paramètres et initier des fonctions d'application et de service dans le système, sont appelées «données d'application».

3 Definitions and abbreviations

For the purposes of this part of IEC 60880, the terms and definitions given in IEC 60880 and in the IAEA Safety Series No. 50-C-D, as well as the following terms and definitions apply.

NOTE 1 "Shall", "should" and "may" are used in accordance with IEC conventions.

NOTE 2 For clarification of the terms **error**, **fault**, **failure** and **signal trajectory** the following example is given.

If a person or process makes an **error** in producing something, this will result in a **fault** in the product. When the product is used, it may be satisfactory, or it may fail, if the **fault** is not corrected. If the use challenges the **fault**, the product will fail if no other defence prevents the **failure**. A **failure** is due to both a **fault** and a challenge, with no other defence operating. For software, a challenge to a **fault** is provided by a **signal trajectory.**

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

3.2
application function
function of an I&C system that performs a task related to the process being controlled rather than to the functioning of the system itself
[derived from 2.1 of IEC 60880]

3.3
channel
separate path along which information flows through a redundant or distributed system. That path may also contain redundancy

3.4
common cause failure (CCF)
failure which is the result of one or more events, causing coincident failures of two or more separate channels in a multiple channel system or in multiple systems, leading to system(s) failure
[3.6.10 of IEC 61508-4, modified]

NOTE 1 Depending on the context, a CCF may be considered at the system level or at the level of the systems which constitute a safety group.

NOTE 2 See definition of failure (3.8).

3.5
data
representation of information or instructions in a manner suitable for communication, interpretation, or processing by computers

[adapted from IEEE 610]

NOTE Data which are required to define parameters and to instantiate application and service functions in the system are called "application data".

3.6
diversité
existence de deux ou plusieurs manières différentes d'atteindre un objectif donné. La diversité est en particulier assurée comme moyen de défense contre une défaillance de mode commun. Elle peut être réalisée par la mise en oeuvre des systèmes physiquement différents les uns des autres ou par une diversité fonctionnelle, dans laquelle des systèmes similaires réalisent l'objectif spécifié de manière différente
[définition adaptée de la CEI 61226]

3.7
analyse dynamique
processus consistant à évaluer un système ou un composant sur la base de son comportement pendant l'exécution. S'oppose à l'analyse statique (voir IEEE 610)

3.8
défaillance
une défaillance survient lorsque le service réalisé s'écarte du service voulu

NOTE Une défaillance est le résultat d'un défaut matériel, d'un défaut logiciel, d'un défaut système, ou d'une erreur d'exploitation ou de maintenance, et de la trajectoire de signal associée qui génère la défaillance.

3.9
défaut
anomalie d'un composant matériel ou logiciel ou système. Les défauts sont divisés en défauts aléatoires (par exemple la conséquence d'une usure du matériel), et défauts systématiques (par exemple introduits dans la conception); pour le logiciel, ceux-ci comprennent les erreurs de codage et les erreurs de spécification

NOTE Un défaut (et en particulier un défaut de conception) peut rester non détecté dans une partie du système jusqu'à ce que des conditions spécifiques ou une trajectoire de signal affectant cette partie du système soient telles que le résultat produit ne soit pas conforme à la fonction désirée. Cela entraîne une défaillance de cette partie du système.

3.10
diversité fonctionnelle
application de la diversité au niveau fonctionnel (par exemple, actionnement d'un déclenchement sur la limite de pression et sur la limite de température)

3.11
fonctions, et systèmes et matériels associés (FSE)
les fonctions sont des actions effectuées dans un but ou afin de réaliser un objectif. Les systèmes et équipements associés sont les ensembles de composants et les composants eux-mêmes qui sont employés pour remplir les fonctions (voir l'article 3 de la CEI 61226)

3.12
erreur (ou faute) humaine
action humaine ou procédure produisant un résultat indésirable

3.13
bibliothèque
ensemble d'éléments logiciels connexes contenus dans un fichier unique, mais sélectionnés individuellement pour inclusion dans le produit logiciel final

3.14
logiciel à n versions ou multi-versions
ensemble de programmes différents, appelés versions, développé pour satisfaire une prescription commune et des tests communs d'acceptation. L'exécution simultanée et indépendante de ces versions se déroule, en général sur des matériels redondants. Des entrées identiques dans des systèmes de tests ou bien des entrées correspondantes dans des systèmes redondants sont utilisées. Une stratégie prédéterminée telle que le vote est utilisée pour choisir parmi des sorties contradictoires de versions différentes

3.6
diversity
existence of two or more different ways or means of achieving a specified objective. Diversity is specifically provided as a defence against common mode failure. It may be achieved by providing systems that are physically different from each other, or by functional diversity, where similar systems achieve the specified objective in different ways (see clause 3 of IEC 61226)

3.7
dynamic analysis
process of evaluating a system or component based on its behaviour during execution. In contrast to static analysis (see IEEE 610)

3.8
failure
failure occurs when the delivered service deviates from the intended service

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

3.9
fault
defect in a hardware, software, or system component. Faults are subdivided into random faults (for example, resulting from wearing out of hardware), and systematic faults (for example, introduced in the design), and for software these include coding errors and specification errors

NOTE A fault (notably a design fault) may remain undetected in a part of the system until specific conditions or signal trajectories affecting that part of the system, are such that the result produced does not conform to the intended function. This results in a failure of that part of the system.

3.10
functional diversity
application of the diversity at the functional level (for example, to have trip activation on both pressure and temperature limit)

3.11
functions, and associated systems and equipment (FSE)
functions are carried out for a purpose or to achieve a goal. The associated systems and equipment are the collections of components and the components themselves that are employed to achieve the functions (see clause 3 of IEC 61226)

3.12
human error (or mistake)
human action or a procedure that produces an unintended result

3.13
library
collection of related software elements that are grouped together, but which are individually selected for inclusion in the final software product

3.14
n-version or multi-version software
set of different programs, known as versions, developed to meet a common requirement and common acceptance test. Concurrent and independent execution of these versions takes place, generally in redundant hardware. Identical inputs in test systems or corresponding inputs in redundant systems are used. A predetermined strategy such as voting is used to decide between conflicting outputs in different versions

3.15
logiciel système opérationnel
logiciel fonctionnant sur le processeur cible, par exemple pilotes et services d'entrée/sortie, gestion des interruptions, gestion de l'ordonnancement, pilotes de communication, bibliothèques orientées application, diagnostic en ligne, redondance et gestion des modes de marche dégradée

3.16
événements initiateurs hypothétiques (PIE)
événements qui entraînent des incidents de fonctionnement prévus ou des situations accidentelles et leurs combinaisons plausibles

[AIEA 50-C-D]

3.17
logiciel prédéveloppé (LPD)
logiciel existant, disponible dans le commerce ou en tant que produit protégé par des droits de propriété, dont l'utilisation est envisagée dans un système informatique

3.18
réutilisable
s'applique à un module logiciel qui peut être utilisé dans plusieurs programmes d'un ordinateur ou d'un système informatique (voir IEEE 610)

3.19
trajectoire de signal
l'historique temporel de tous les situations des équipements, états d'un matériel, signaux d'entrée et entrées de l'opérateur qui déterminent les sorties d'un système

3.20
version logicielle
nouvelle édition d'un produit logiciel consécutive à une modification ou à une correction d'un produit logiciel précédent

3.21
spécification
document qui spécifie de manière complète, précise et vérifiable les prescriptions, la conception, le comportement ou autres caractéristiques d'un système ou composant et, souvent, les procédures permettant de déterminer si ces dispositions ont été satisfaites (voir IEEE 610)

NOTE Il existe différents types de spécifications, par exemple les spécifications des exigences de logiciels et les spécifications de conception.

3.22
analyse statique
processus d'évaluation d'un système ou d'un composant basé sur sa forme, sa structure, son contenu ou sa documentation. S'oppose à l'analyse dynamique

3.23
logiciel de support
logiciel d'aide au développement, au test ou à la maintenance d'autres logiciels, comme par exemple les compilateurs, les générateurs de codes, les simulateurs, le diagnostic hors ligne, les initialiseurs (voir 4.2)

3.24
logiciel système
logiciel conçu pour un système programmé ou une famille spécifique de systèmes programmés afin de faciliter l'exploitation et la maintenance du système programmé et des programmes associés, par exemple systèmes d'exploitation, compilateurs, utilitaires. Le logiciel système se compose en général du logiciel système opérationnel et du logiciel de support

3.15
operational system software
software running on the target processor during operation, such as input/output drivers and services, interrupt management, scheduler, communication drivers, application-oriented libraries, on-line diagnostic, redundancy and graceful degradation management

3.16
postulated initiating event (PIE)
identified events that lead to anticipated operational occurrences or accident conditions and their consequential failure effects

[IAEA 50-C-D]

3.17
pre-developed software (PDS)
software which already exists is available as commercial or proprietary product and is being considered for use in a computer-based system

3.18
reusable
pertaining to a software module that can be used in more than one computer program or software system (derived from IEEE 610)

3.19
signal trajectory
time histories of all equipment conditions, internal states, input signals, and operator inputs which determine the outputs of a system

3.20
software version
instance of a software product derived by modification or correction of a preceding software product

3.21
specification
document that specifies, in a complete, precise, verifiable manner, the requirements, design behaviour or other characteristics of a system or component and, often, the procedures for determining whether these provisions have been satisfied (see IEEE 610)

NOTE There are different types of specifications, for example, software requirements specification or design specification.

3.22
static analysis
process of evaluating a system or component based on its form, structure, content or documentation. In contrast to dynamic analysis

3.23
support software
software that aids in the development, test, or maintenance of other software such as compilers, code generators, simulators, off-line diagnostic, initializers (see 4.2)

3.24
system software
software designed for a specific computer system or family of computer systems to facilitate the operation and maintenance of the computer system and associated programs, for example, operating systems, compilers, utilities. System software is usually composed of operational system software and support software

4 Prescriptions et recommandations

Dans la suite le terme «exigences» est utilisé pour désigner à la fois les exigences et les recommandations. La distinction apparaît au niveau de chaque paragraphe, où les exigences sont exprimées par «doit» et les recommandations par «il convient que».

4.1 Moyens de défense contre les défaillances logicielles de cause commune

Ce paragraphe émet des exigences en matière de moyens de défense contre les défauts de conception et codage qui peuvent amener à des défaillances de cause commune (CCF) de fonctions classées dans la Catégorie A, conformément à la CEI 61226.

4.1.1 Introduction

Une CCF peut intervenir dans les systèmes et équipements de l'architecture d'I&C qui réalisent différentes lignes de défense pour le même PIE (voir 5.3.1 de la CEI 61513). Le logiciel par lui-même n'a pas une CCF. Une CCF est à mettre en relation avec des défaillances qui ont leur origine dans des erreurs des exigences fonctionnelles, de la conception du système ou du logiciel.

Pour toutes les activités de sûreté, qu'elles soient relatives à l'organisation, au comportement ou à la conception, l'AIEA demande d'appliquer la défense en profondeur (voir 204 de la 50-C-D de l'AIEA), pour assurer que ces activités ont des recouvrements, de façon que si une défaillance intervient dans un sous-système, elle puisse être compensée ou corrigée dans le système total.

Le critère de défaillance unique (voir 329 à 336 de la 50-C-D de l'AIEA) stipule que l'ensemble des systèmes de sûreté doivent pouvoir remplir leur mission malgré une défaillance aléatoire unique pouvant intervenir n'importe où dans cet ensemble.

Les défauts logiciels sont des défauts systématiques et non des défauts aléatoires. Par conséquent, le critère de défaillance unique ne peut pas être appliqué à la conception d'un système de la même manière qu'il a été appliqué pour le matériel. Ainsi, lorsque le concept de défense en profondeur est appliqué, il convient de prendre en compte les éventuels effets des CCF logicielles dans chacune des couches de défense et entre couches redondantes. Il convient d'adopter des contre-mesures appropriées pendant le processus de développement et dans le processus d'évaluation, par exemple

a) dans le développement, la vérification et la validation de chacune des couches de défense; et

b) dans l'évaluation de l'indépendance et de la diversité des couches de défense redondantes.

L'utilisation de la diversité est un des moyens pour améliorer la fiabilité de certains systèmes et pour réduire le risque de certaines CCF (voir 337 à 339 de la 50-C-D de l'AIEA).

4.1.1.1 Prescriptions de la CEI 60880

Le paragraphe 4.2 de la CEI 60880 stipule que les prescriptions de fiabilité du système doivent servir de toile de fond à la description de la configuration du système informatique et renvoie à A.2.2 de la CEI 60880. Ce paragraphe donne une liste des facteurs pertinents (y compris la défense en profondeur, le repli en mode de fonctionnement dégradé, la gestion des défaillances, la diversité fonctionnelle et la diversité logicielle, la séparation et décomposition en modules, le découplage, la séparation logique) qui peuvent être utilisés pour éviter les CCF, mais ne traite pas ces facteurs de manière spécifique. Par conséquent, le paragraphe 4.1 de cette partie de la CEI 60880 spécifie comment déterminer les CCF potentielles provoquées par les logiciels ainsi que les techniques pour éviter l'apparition des CCF.

4 Requirements and recommendations

In the following, the term "requirements" is used as an inclusive term for both requirements and recommendations. The distinction appears at the level of the individual clauses where requirements are expressed by "shall" and recommendations by "should".

4.1 Defences against common cause failure due to software

This subclause provides requirements for defences against software design and coding faults which can lead to common cause failures (CCF) of functions classified as category A according to IEC 61226.

4.1.1 Introduction

CCF may occur in the I&C architecture's systems and equipment implementing different lines of defence against the same PIE (see 5.3.1 of IEC 61513). Software by itself does not have a CCF mode. CCF is related to system failures arising from faults in the functional requirements, system design, or in the software.

Defence in depth is required by the IAEA (see 204 of IAEA 50-C-D) to be applied to all safety activities, whether organizational, behavioural or design related, to ensure that there are overlapping defences so that if a failure should occur in a subsystem, it would be compensated for, or corrected in the integral system.

The single-failure criterion (see 329 to 336 of IAEA 50-C-D) requires that the assembly of safety systems have the ability to meet its purpose despite a single random failure assumed to occur anywhere in the assembly.

Software faults are systematic, not random, faults and, therefore, the single-failure criterion can not be applied to the software design of a system in the same manner as it has been applied for hardware. When the defence-in-depth concept is applied, possible effects of CCF due to software inside each defence layer and between redundant layers have to be considered and appropriate counter-measures have to be adopted throughout the development process and in the evaluation processes, for example,

a) in the development, verification and validation of each individual defence layer; and

b) in the evaluation of the independence and diversity of redundant defence layers.

A means of enhancing the reliability of some systems and reducing the potential for certain CCFs is the use of diversity (see 337 to 339 of IAEA 50-C-D Rev 1).

4.1.1.1 Requirements of IEC 60880

Subclause 4.2 of IEC 60880 requires the system reliability requirements to be the background to the description of the computer system configuration, and refers to A.2.2 of IEC 60880. This latter subclause indicates relevant factors (including defence in depth, graceful degradation, management of failures, functional diversity, software diversity, spatial separation and modularization, decoupling, logical separation) that might be used to avoid CCF, but does not specifically address all these factors. Therefore, 4.1 of this part of IEC 60880 specifies the way to assess the potential for CCF caused by software and also specifies techniques to avoid the occurrence of CCF.

4.1.1.2 Principes directeurs de la défense contre les CCF logicielles

Les principes directeurs de la défense contre les CCF logicielles énoncent que tout défaut logiciel restera dans le système ou la voie concernés jusqu'à ce qu'il soit détecté et corrigé, et qu'il causera une défaillance si une trajectoire de signal préjudiciable la sollicite. Si deux systèmes ou canaux ou plus, qui réalisent différentes lignes de défense pour le même PIE (voir 5.3.1.5 de la CEI 61513), contiennent le défaut et sont sollicités par des trajectoires de signal spécifiques au cours d'une période sensible, les deux (ou tous/toutes) systèmes ou voies connaîtront une défaillance. Une description plus détaillée de ces états est donnée à l'article A.1.

Il convient donc de prendre en compte le risque potentiel de CCF logicielle lors de la conception. Si des conditions de CCF hypothétiques peuvent être prévues, des changements de conception et des fonctionnalités de défense, y compris la diversité logicielle, peuvent être nécessaires pour assurer la protection contre les CCF logicielles.

Le degré d'amélioration des défenses contre les CCF et les améliorations de la fiabilité qui peuvent être obtenues grâce à la diversité ne peuvent pas être quantifiées. Le concepteur utilisera son meilleur jugement sur la base d'une évaluation qualitative de la fiabilité que peut assurer le logiciel.

Les erreurs humaines intervenant avant le début de la conception logicielle donnent lieu à des défauts de prescriptions et à des défaillances potentielles du système contre lesquelles le seul génie logiciel ne peut prémunir. La défense contre ces types de CCF est traitée au niveau système en 5.3.1.5 de la CEI 61513.

Des erreurs humaines intervenant pendant le processus de génie logiciel peuvent donner lieu à des défauts logiciels et à des défaillances potentielles. Lorsque de tels défauts provoquent la défaillance de plus d'une ligne de défense, les défaillances sont considérées comme des CCF logicielles.

4.1.2 Conception du logiciel pour éviter les CCF

Le moyen de défense élémentaire et le plus important contre les CCF logicielles est de produire des logiciels de la plus haute qualité, c'est-à-dire contenant le moins d'erreurs possible. L'étendue de la couverture des moyens d'autocontrôle, tels que la surveillance de plausibilité des données, le contrôle d'échelle des paramètres, la temporisation cyclique, etc. (voir 4.8, 5.1 et A.2.8 de la CEI 60880), constituent un autre facteur important pour limiter les CCF logicielles potentielles.

Des exigences pour obtenir une haute fiabilité du logiciel sont données dans la CEI 60880 et dans les paragraphes suivants de la présente norme.

L'utilisation de méthodes éprouvées de génie logiciel avec des outils d'aide au développement et à la vérification du logiciel peuvent réduire le nombre de décisions humaines dans la conception et peuvent ainsi réduire le nombre de défauts dans le logiciel produit.

4.1.3 Sources et effets des CCF logicielles

4.1.3.1 Une analyse documentée des CCF logicielles potentielles doit être effectuée et documentée au niveau du système et/ou au niveau de l'architecture globale des systèmes d'I&C importants pour la sûreté.

NOTE 1 Des exigences sur l'architecture I&C sont données en 5.3.1 de la CEI 61513.

NOTE 2 Des exigences sur l'architecture des systèmes individuels d'I&C sont données en 6.1.2 de la CEI 61513.

4.1.1.2 Rationale for defence against CCF due to software

The rationale for defence against software faults is that any software fault will remain in the system or channel concerned until detected and corrected, and can cause failure if a specific signal trajectory challenges it. If two or more systems or channels implementing different lines of defence for the same PIE (see 5.3.1.5 of IEC 61513) contain the fault, and are exposed to specific signal trajectories within a sensitive time period, both (or all) systems or channels can fail, which is called a CCF. A more detailed description of these conditions is given in clause A.1.

The potential for CCF due to software should therefore be considered during design. If postulated conditions of CCF can be foreseen, design changes and defence features, including software diversity, may be needed for protection against CCF due to software.

The degree of improvement of defence against CCF and improvement in reliability that can be achieved by diversity cannot be quantified. Judgement is required based on an evaluation of the qualitative reliability which the software can achieve.

If human errors are made before software design starts, they may lead to faults of requirements and potential system failures against which software engineering alone cannot provide a defence. Defence against such CCF is discussed at the system level in 5.3.1.5 of IEC 61513.

If human errors are made during the software engineering process, they may lead to software faults and potential system failures. Where such faults lead to the failure of more than one line of protection the failures are considered to be CCFs due to software.

4.1.2 Design of software against CCF

The basic, and most important, defence against CCF due to software is to produce software of the highest quality, i.e. as error-free as possible. The extent of coverage of self-monitoring features, such as for data plausibility, parameter range checking, and loop timing etc. as addressed by 4.8, 5.1 and A.2.8 in IEC 60880 is a further important factor in limiting the potential for CCF due to software.

Requirements to achieve highly reliable software with self-monitoring features are given in IEC 60880 and the following paragraphs of this standard.

The use of well-developed software engineering methods with software tool support for software development and verification can help to reduce the number of human design decisions and so potentially reduce the number of faults in the developed software.

4.1.3 Sources and effects of CCF due to software

4.1.3.1 An analysis of the potential for CCF due to software shall be performed and documented at the system level and/or at the level of the total I&C architecture of the I&C systems important to safety of the NPP.

NOTE 1 Requirements on the I&C architecture are given in 5.3.1 of IEC 61513.

NOTE 2 Requirements on the architecture of the individual I&C systems are given in 6.1.2 of IEC 61513.

4.1.3.2 Il convient que l'analyse comprenne les étapes suivantes:

a) identification des composants logiciels utilisés dans le système ou dans l'architecture du contrôle commande;

b) analyse des CCF potentielles dues à ces composants dans le cadre du système ou de l'architecture du contrôle commande;

c) analyse des effets possibles de ces CCF.

NOTE L'analyse des effets potentiels des défauts ne supprime pas le besoin d'effectuer les activités de vérification et validation requise par la CEI 60880. Le but d'une telle analyse est de mettre en évidence toutes les faiblesses dans la conception, afin d'y initier des changements et/ou d'améliorer la confiance dans la conception du logiciel.

4.1.3.3 Les modules communs utilisés dans plusieurs systèmes doivent être identifiés et l'assurance de la fiabilité de ces modules doit être démontrée. Des méthodes pour mener cette démonstration sont donnés à l'article A.4.

4.1.3.4 Les données transmises à l'intérieur du système informatique ou entre différents systèmes informatiques doivent être identifiées. Une analyse doit être effectuée pour déterminer si des données erronées peuvent provoquer une CCF dans les processeurs ou les systèmes en réception.

4.1.3.5 La possibilité que, dans certaines conditions propres à la centrale, des mêmes logiciels implantés dans différents matériels soient soumis simultanément à des trajectoires de signaux identiques et qu'un même défaut logiciel apparaisse dans plusieurs canaux ou chemins fonctionnels, doit être évaluée.

NOTE Les défaillances peuvent être provoquées par des trajectoires de signaux qui n'ont pas été pris en compte lors de la conception, de la vérification et de la validation des logiciels des canaux individuels ou systèmes.

4.1.3.6 Les activités de maintenance du logiciel constituent une source potentielle de CCF. Il convient que la procédure de modification du logiciel ou des données donne l'assurance que ces types de défauts ne sont pas introduits.

4.1.3.7 Une analyse des sources potentielles de CCF logicielles doit être faite et documentée dans le cadre du jugement sur la défense contre les CCF propre à la conception de l'architecture I&C (voir 5.3.3 de la CEI 61513).

4.1.3.8 Si l'analyse identifie une menace inacceptable résultant de CCF provoquées par le logiciel, la conception du logiciel ou de l'architecture du contrôle commande doit être améliorée. Des méthodes pour la réalisation de défenses contre les CCF sont données à l'article A.3, et des méthodes pour la mise en œuvre de la diversité sont données à l'article A.5.

4.1.4 Mise en oeuvre de la diversité

4.1.4.1 Il convient que la mise en oeuvre de la diversité utilise des systèmes indépendants avec diversité fonctionnelle. Si la diversité fonctionnelle n'est pas appropriée ou possible, il convient de prendre en considération la diversité des systèmes, la diversité des caractéristiques des logiciels et la diversité des approches de conception. Les caractéristiques importantes sont données dans l'article A.5. Les techniques utilisées pour la défense contre les CCF doivent être documentées et justifiées par rapport à l'analyse qui a été effectuée.

4.1.4.2 Au niveau du logiciel, il convient que la défense contre les CCF soit basée sur un ensemble de techniques approprié, comme:

a) des garanties sur la diversification des conditions d'exploitation du logiciel;

b) des protections contre les erreurs et la propagation des défaillances;

4.1.3.2 The analysis should include the following steps:

a) identification of the software components used in the system or I&C architecture;

b) analysis of the potential CCF due to these components within the system or the I&C architecture;

c) analysis of the possible effects of these CCF.

NOTE Performing an analysis of the potential effects of faults does not obviate the need to perform verification and validation activities as required by IEC 60880. The purpose of such an analysis is to reveal any weaknesses in the design, and hence initiate changes to the design, and/or to improve confidence in the software design.

4.1.3.3 If common modules are used in more than one system, they shall be identified and assurance of the reliability of such common modules shall be assessed. Methods supporting the demonstration of correctness are given in clause A.4.

4.1.3.4 Data transmitted inside a computer-based system or between computer-based systems shall be identified. An analysis shall be performed to determine if faulty data can lead to a CCF in receiving computers or systems.

4.1.3.5 The potential for plant conditions to subject the same software running in different hardware to identical and simultaneous signal trajectories and hence reveal the same software fault in several channels or functional paths shall be assessed.

NOTE Failures may be caused by signal trajectories which were not considered during the design, verification and validation of the software of the individual channels or systems.

4.1.3.6 Software maintenance activities have the potential to cause CCF and the processes used for software or data change assessment should provide assurance that such faults are not introduced.

4.1.3.7 The analysis of potential CCF due to software shall be performed and documented as part of the assessment of the defence against CCF of the design of the I&C architecture (see 5.3.3 of IEC 61513).

4.1.3.8 If the analysis identifies an unacceptable threat arising from CCF due to software then the design of the software or of the I&C architecture shall be improved. Methods supporting the implementation of CCF defences are given in clause A.3 and methods supporting the implementation of diversity features are given in clause A.5.

4.1.4 Implementation of diversity

4.1.4.1 Implementation of diversity should use independent systems with functional diversity. If functional diversity is not appropriate or possible, the use of system diversity, diverse software features and diverse design approaches should be considered. Features of importance are given in clause A.5. The techniques chosen for defence against CCF shall be documented and justified according to the analysis made.

4.1.4.2 At the software level, defences against CCF should be based on an appropriate selection of techniques, such as

a) guarantee of diversified operational conditions of the software;

b) defences against error and failure propagation;

c) la réduction des effets négatifs des CCF;

d) l'utilisation de logiciels diversifiés répondant à des spécifications différentes, pour des réalisations différentes de la même exigence fonctionnelle.

NOTE 1 Il convient de prendre en considération la possibilité de différencier les méthodes de conception et implémentation mais ceci n'est pas une exigence.

NOTE 2 La technique de la N programmation n'est pas recommandée.

4.1.5 Pondération des inconvénients et des avantages liés à l'utilisation de la diversité

Si la diversité logicielle est utilisée et affirmée, il convient que les inconvénients et avantages sur la fiabilité globale du logiciel soient justifiés sur la base de l'analyse ci-dessus, et documentés (voir 339 de la 50-C-D de l'AIEA, et 4.6.2 de la 50-SG-D11 de l'AIEA). Les avantages, inconvénients et aspects liés à la justification sont donnés à l'article A.6.

4.2 Outils logiciels pour le développement de logiciels

4.2.1 Introduction

Le présent paragraphe développe les prescriptions existantes de la CEI 60880 pour les outils logiciels utilisés dans le développement de logiciels destinés aux calculateurs des systèmes de sûreté des centrales nucléaires.

4.2.1.1 Prescriptions de la CEI 60880 liées aux outils logiciels

Les prescriptions de la CEI 60880 spécifiquement applicables à l'utilisation, la vérification et l'évaluation des outils logiciels sont listées dans l'annexe B de la présente norme.

4.2.1.2 Utilisation des outils logiciels

L'utilisation des outils logiciels appropriés peut augmenter l'intégrité du processus de développement du logiciel, et donc la fiabilité du produit logiciel, en réduisant le risque d'introduction d'erreurs pendant le processus. L'utilisation d'outils peut également présenter des avantages économiques car le temps et le travail nécessaires à la production du logiciel peuvent être réduits. Les outils peuvent être utilisés pour contrôler automatiquement le respect des règles et normes de construction, générer des enregistrements adéquats et une documentation cohérente aux formats standards, et permettre le contrôle des modifications. Les outils peuvent également réduire les travaux de test et permettre la tenue automatique de journaux. Les outils peuvent aussi être nécessaires car requis par une méthode spécifique de développement.

Les outils sont particulièrement puissants lorsqu'ils sont définis pour fonctionner conjointement. Il convient de prendre garde à ne pas attendre des outils qu'ils réalisent des tâches qui dépassent leurs capacités. En d'autres termes, ils ne peuvent pas remplacer l'homme lorsqu'il convient de faire appel à son jugement. Dans certains cas, le soutien par des outils est mieux adapté que l'automatisation complète du processus. Lors de la sélection d'un outil, il convient de faire la balance entre les avantages et les risques liés à son utilisation et les avantages et les risques liés à sa non-utilisation. Le principe important est de choisir des outils qui limitent les possibilités d'introduction d'erreurs mais qui maximisent la capacité à les détecter.

Les outils qui entrent dans le cadre de cette partie de la CEI 60880 sont ceux utilisés pour permettre la capture des prescriptions et ceux utilisés pour permettre la transformation des prescriptions en codes et en données système finals (il peut y avoir un grand nombre d'étapes intermédiaires). Cette partie recouvre également les outils utilisés pour permettre de procéder directement à la vérification, à la validation et au test, les outils utilisés pour la préparation et le contrôle des données d'application (voir 4.2.3.5), et enfin les outils utilisés pour la gestion et le contrôle des processus et des produits concernés par le développement des logiciels.

c) reduction of the negative effects of CCF;

d) use of software diversified by different specifications for different implementations of the same functional requirement.

NOTE 1 Differences in design and implementation methods should be considered for inclusion but are not required.

NOTE 2 N-version programming is not recommended.

4.1.5 Balance of drawbacks and benefits connected with the use of diversity

If diverse software is used and claimed, the drawbacks and benefits on the overall reliability of the software should be justified on the basis of the above analysis, and documented (see 339 of IAEA 50-C-D, and 4.6.2 of IAEA 50-SG-D11). Potential benefits, drawbacks and justification aspects are given in clause A.6.

4.2 Software tools for the development of software

4.2.1 Introduction

This subclause expands on the existing requirements of IEC 60880 for software tools used in the development of software for computers in safety systems of nuclear power plants.

4.2.1.1 IEC 60880 requirements for software tools

The IEC 60880 requirements that specifically apply to the use, verification and assessment of software tools are listed in annex B of this standard.

4.2.1.2 Rationale for the use of software tools

The use of appropriate software tools can increase the integrity of the software development process, and hence software product reliability, by reducing the risk of introducing faults in the process. The use of tools can also have economic benefits as they can reduce the time and human effort required to produce software. Tools can be used to automatically check for adherence to rules of construction and standards, to generate proper records and consistent documentation in standard formats, and to support change control. Tools can also reduce the effort required for testing and to maintain automatic logs. Tools can also be necessary because a specific development methodology requires their use.

Tools are most powerful when they are defined to work co-operatively with each other. Care should be taken not to require tools to undertake tasks beyond their capability, for example, they cannot substitute humans when judgement is involved. In some cases, tool support is more appropriate than complete automation of the process. When selecting a tool, the benefits and risk of using a tool must be balanced against the benefits and risk of not using a tool. The important principle is to choose tools that limit the opportunity for making errors and introducing faults, but maximize the opportunity for detecting faults.

Tools within the scope of this part of IEC 60880 include those used to support the capture of requirements and those used to support the transformation of requirements into the final system code and data (there may be many intermediate steps). The scope also includes those tools used to directly support the performance of verification, validation and testing, tools for the preparation and control of application data (see 4.2.3.5), and tools for the management and control of the processes and products involved in the software development.

Les outils hors-ligne, employés pour calculer les variables importantes utilisées lors de la conception et de l'analyse des équipements importants pour la sûreté, sortent du cadre de la CEI 60880 et de la CEI 60880-2, de même que les traitements de texte, les outils de gestion de projet et autres outils bureautiques et de soutien administratif utilisés pour des tâches non directement concernées par le développement logiciel.

4.2.2 Sélection des outils

4.2.2.1 Les outils utilisés pour le développement de logiciels des systèmes de sûreté doivent être sélectionnés pour aider le processus d'ingénierie du logiciel. Les critères et le processus de sélection à suivre sont décrits en 4.2.3.1. Les limites d'application des outils doivent être identifiés et documentés. Les outils et leur sortie ne doivent pas être utilisés en dehors de leurs limites déclarées d'application sans justification préalable.

Les outils utilisés pour le développement des logiciels des systèmes de sûreté doivent être vérifiés et évalués à un niveau adéquat par rapport aux prescriptions de fiabilité de l'outil, au type d'outil (voir les points a) à e) de 4.2.2.2) et au potentiel de l'outil à introduire des défauts.

Les outils doivent présenter une fiabilité suffisante pour assurer qu'ils ne mettent pas en cause l'intégrité du produit final. Par exemple, un outil peut affecter défavorablement le développement de logiciels en introduisant des erreurs, en produisant une sortie corrompue, ou en ne détectant pas un défaut déjà présent.

Les principes de défense en profondeur et de diversité adoptés pour l'architecture du contrôle commande peuvent être pris en compte lors de la sélection des outils afin de réduire les besoins de fiabilité des outils pris individuellement.

4.2.2.2 Le niveau de vérification et d'évaluation requis pour un outil dépend également de la classe ou du type d'outil et de la possibilité ou non de vérifier ou de valider complètement la sortie de l'outil. Les classes d'outils sont:

a) les outils de transformation tels que les générateurs de code, les compilateurs et ceux qui transforment un texte ou un diagramme d'un niveau d'abstraction à un autre, en général inférieur;

b) les outils de vérification et de validation tels que les logiciels d'analyse statique, les contrôleurs de couverture des tests, les logiciels d'aide à la démonstration de théorèmes et les simulateurs;

c) les outils de diagnostic utilisés pour la maintenance et l'observation du logiciel pendant l'exploitation;

d) les outils d'infrastructure tels que les systèmes supports de développement;

e) les outils de contrôle de la configuration tels que les outils de contrôle de version.

4.2.3 Prescriptions applicables aux outils

Les prescriptions applicables aux outils sont présentées comme suit:

a) environnement de génie logiciel;

b) qualification des outils;

c) gestion de la configuration des outils;

d) traducteurs/compilateurs;

e) outils pour données d'application;

f) automatisation des tests.

Off-line tools, used to calculate important variables used during the design and analysis of equipment important to safety, are considered beyond the scope of IEC 60880 and IEC 60880-2. Word processors, project management tools, and other office administration tools used to support tasks not directly concerned with software development are also not within its scope.

4.2.2 Selection of tools

4.2.2.1 The tools used to develop software for safety systems shall be selected to support the software engineering process. The criteria and process of tool selection to be followed are described in 4.2.3.1. The limits of applicability of all tools shall be identified and documented. The tools and their output shall not be used outside their declared limits of application without prior justification.

The tools used in the development of software in safety systems in nuclear power plants are required to be verified and assessed to a level consistent with the tool reliability requirements, the type of tool (see items a) to e) of 4.2.2.2) and the potential of the tool to introduce faults.

Tools shall have sufficient reliability to ensure that they do not jeopardize the reliability of the end product. For example, a tool could adversely affect the development of software by introducing errors, by producing a corrupted output, by failing to detect a fault that is already present.

Principles of defence in depth and diversity adopted for I&C architecture may be considered when selecting tools, in order to reduce the reliability requirements on individual tools.

4.2.2.2 The level of verification and assessment required for a tool also depends on the type of tool and whether the output of the tool can be fully verified or validated. Types of tools are:

a) transformation tools such as code generators, compilers, and those that transform text or a diagram at one level of abstraction into another, usually lower, level of abstraction;

b) verification and validation tools such as static code analysers, test coverage monitors, theorem proving assistants, and simulators;

c) diagnostic tools used to maintain and monitor the software under operating conditions;

d) infrastructure tools such as development support systems;

e) configuration control tools such as version control tools.

4.2.3 Requirements for tools

Requirements for tools are presented by topic:

a) software engineering environment;

b) tool qualification;

c) tool configuration management;

d) translators/compilers;

e) application data tools;

f) automation of testing.

4.2.3.1 Environnement de génie logiciel

4.2.3.1.1 Il convient que les outils soient utilisés pour supporter tous les aspects du cycle de vie du logiciel lorsque leur utilisation comporte des avantages et lorsque les outils existent. L'analyse de l'environnement de génie logiciel et des processus de développement doit être effectuée pour déterminer la stratégie de soutien par les outils. Il convient que les résultats de l'analyse soient documentés. Si les outils n'existent pas, il peut être nécessaire d'envisager le développement d'outils nouveaux.

Exemples de processus et opérations qui peuvent tirer avantage d'un soutien par des outils:

a) la production et la vérification de documents de spécification, conception et codage (voir annexe C);

b) les outils fonctionnant sur le langage ou sur un sous-ensemble du langage (voir 4.2.3.4);

c) la préparation, la vérification, la validation et la gestion des données applicatives (voir 4.2.3.5);

d) l'automatisation des tests (voir 4.2.3.6).

4.2.3.1.2 Il convient que les critères de sélection et d'évaluation de l'environnement de génie logiciel soient développés et classés par ordre de priorité afin de permettre des compromis avant l'utilisation. Il convient que les critères soient structurés selon les caractéristiques de qualité du logiciel: fonctionnalité, fiabilité, usabilité, efficacité, maintenabilité, et portabilité, comme définies dans l'ISO/CEI 9126. Les critères peuvent inclure d'autres sujets, comme: l'effort nécessaire pour obtenir l'agrément des autorités, les ressources nécessaires pour utiliser un outil, la rigueur du programme qualité sous lequel l'outil a été développé, l'historique du fournisseur de l'outil et les alternatives à l'utilisation de l'outil.

4.2.3.1.3 Le soutien par les outils de l'environnement de génie logiciel doit être analysé et documenté afin de déterminer:

a) comment chacun des processus est, ou n'est pas, soutenu par des outils;

b) l'identification précise des outils (par exemple nom et numéro de version), et si possible leur configuration;

c) comment chacun des outils doit être utilisé dans le cadre du projet (c'est-à-dire la classe de l'outil);

d) comment la sortie de chaque outil doit être vérifiée et/ou validée par rapport aux entrées;

e) comment les autres outils ou processus atténuent les conséquences d'une erreur dans l'outil, y compris l'atténuation d'erreurs potentielles pendant la production et la préparation des données pour utilisation en ligne;

f) quelles sont les interfaces des outils avec d'autres outils, à savoir des outils peuvent être nécessaires pour utiliser, traiter et livrer de l'information partagée par d'autres outils ou faisant partie d'une base de données;

g) comment les outils donnent une interface cohérente pour les utilisateurs et pour l'environnement de génie logiciel restant;

h) comment les outils sont adaptés pour les méthodes de génie logiciel sélectionnées;

i) les capacités de détection et de traitement d'erreurs des outils;

j) comment les outils satisfont le contexte d'utilisation comprenant les utilisateurs, le matériel, l'environnement et les tâches des utilisateurs, pour rendre maximale l'efficacité de l'utilisateur et minimiser l'impact des erreurs de l'utilisateur;

k) comment les outils sont protégés contre toute utilisation non autorisée ou non appropriée ou contre des modifications.

4.2.3.1 Software engineering environment

4.2.3.1.1 Tools should be used to support all aspects of the software life cycle where benefits result through their use and where tools are available. Analysis of the software engineering environment and development processes shall be performed to determine the strategy for providing tool support. The results of the analysis should be documented. If tools are not available, the development of new tools may need to be considered.

Examples of processes and operations that can benefit from tool support are:

a) production and checking of specification, design and code (see annex C);

b) tools operating on the language or a subset of the language (see 4.2.3.4);

c) preparation, verification and validation, and management of application data (see 4.2.3.5);

d) automation of testing (see 4.2.3.6).

4.2.3.1.2 Criteria for the selection and evaluation of tools for the software engineering environment should be developed and prioritized to allow trade-offs prior to use. The criteria should be structured by software quality characteristics as defined in ISO/IEC 9126: functionality, reliability, usability, efficiency, maintainability, and portability. The criteria may include other items like licensing effort and resources required to use a tool, the rigour of the quality program under which the tool was developed, vendor tool history, and alternatives to tool use.

4.2.3.1.3 The tool support for the software engineering environment shall be analysed and documented to address:

a) how each process is, or is not, supported by tools;

b) the precise identification of the tools (for example, name, version number) and possibly their configuration;

c) how each tool is to be used within the project;

d) how the output of each tool is to be verified and/or validated against its input;

e) how other tools or processes mitigate the consequences of a fault in the tool, including mitigation of potential errors during production and preparation of data for on-line use;

f) how tools interface with other tools, i.e. tools may be required to use, process, and deliver information shared by other tools or part of a repository;

g) how tools provide a consistent interface to users and to the remainder of the software engineering environment;

h) how tools are suitable for the software engineering methods selected;

i) the error detection and handling capability of tools;

j) how tools satisfy the context of use including users, equipment, environment, and user's tasks, to maximize user effectiveness and minimize the impact of user errors;

k) how tools prevent unauthorized use/misuse or modification.

4.2.3.1.4 La stratégie de maintenance, d'évolution ou de remplacement des outils doit être documentée et justifiée. Cette stratégie fait partie de la stratégie de maintenance du logiciel opérationnel qui doit garantir que le logiciel opérationnel sera maintenu tout au long de son utilisation dans la centrale nucléaire. Elle doit en particulier garantir que le passage à une nouvelle version d'un outil est justifiée et que la nouvelle version est qualifiée de façon appropriée, c'est-à-dire par rapport aux prescriptions de cette norme.

4.2.3.1.5 Il convient de démontrer que les outils utilisés pour fournir une diversification, c'est-à-dire les compilateurs utilisés pour le développement de versions multiples de systèmes de logiciel différents, sont différents. Ceci peut être obtenu en montrant que

a) chaque outil a été fourni par un fournisseur différent (par exemple, un outil peut être développé et un autre acheté dans le commerce); ou
b) les outils ont des langages d'entrée et/ou sortie différents; ou
c) les outils ont des exigences et un processus de conception différents.

4.2.3.2 Qualification des outils

4.2.3.2.1 Les outils doivent être qualifiés selon une stratégie de qualification documentée. La stratégie doit prendre en compte les prescriptions de fiabilité de l'outil et le type d'outil.

4.2.3.2.2 Les prescriptions de fiabilité d'un outil doivent être déterminées compte tenu

a) des conséquences d'une erreur dans l'outil;
b) de la probabilité qu'un outil provoque ou induise des erreurs dans le logiciel de la fonction de sûreté;
c) des autres outils ou processus qui atténuent les conséquences d'une erreur dans l'outil.

NOTE Les principes de défense en profondeur et de diversité peuvent réduire les exigences de fiabilité des outils.

4.2.3.2.3 La stratégie de qualification des outils doit prendre en compte

a) l'analyse du processus de développement des outils et l'historique du fournisseur;
b) l'aptitude de la documentation de l'outil à permettre la vérification de la sortie de l'outil et la facilité d'apprendre;
c) le test ou la validation de l'outil;
d) l'évaluation de l'outil pendant une période d'utilisation;
e) le retour d'expérience sur l'utilisation de l'outil.

NOTE Le paragraphe 4.3 contient des exigences pour l'utilisation de logiciels prédéveloppés qu'il convient de prendre en considération pour la stratégie de qualification des outils.

4.2.3.2.4 Il convient que les sorties de l'outil soient vérifiées systématiquement (par exemple par test, analyse ou comparaison avec les sorties d'outils ayant des fonctionnalités similaires), si les sorties doivent être intégrées dans le logiciel final.

4.2.3.2.5 Si les sorties de l'outil peuvent introduire des défauts dans le logiciel final ne sont pas vérifiées systématiquement, et s'il n'y a pas d'atténuation des effets de défauts de l'outil (par la diversification du processus ou de la conception du système), alors la vérification et l'évaluation de l'outil doivent être réalisées comme décrit en 4.3, ou bien l'outil doit être développé en conformité avec les prescriptions de la CEI 60880. Le processus de qualification peut prendre en compte l'expérience opérationnelle de l'outil lorsque l'utilisation de l'outil a été justifiée dans une utilisation similaire de la même catégorie pour laquelle les conséquences des défaillances sont similaires.

4.2.3.1.4 The maintenance, upgrade or replacement strategy for tools shall be documented and justified. This strategy is a part of the operational software maintenance strategy which shall ensure that the operational software can be maintained throughout its use in the NPP. It shall also ensure that moving to a new version of a tool is justified and the new version of the tool is suitably qualified, i.e. assessed against the requirement of this standard.

4.2.3.1.5 Tools used to provide diversity, i.e. compilers used for the development of multiple-version dissimilar software systems, should be demonstrated to be dissimilar. This may be achieved by showing that

a) each tool was obtained from a different supplier (for example, one tool could be developed and the other tool could be purchased off the shelf); or

b) the tools have different input and/or output languages; or

c) the tools have dissimilar requirements and design processes.

4.2.3.2 Tool qualification

4.2.3.2.1 A tool qualification strategy shall be produced and the tools shall be qualified in accordance with that strategy. The strategy shall consider the reliability requirements of the tool and the type of the tool.

4.2.3.2.2 The reliability requirements of a tool shall be determined considering

a) the consequences of a fault in the tool;

b) the probability that a tool causes or induces faults in the software implementing the safety function;

c) what other tools or processes mitigate the consequences of a fault in the tool.

NOTE Principles of defence in depth and diversity can reduce the reliability requirements on tools.

4.2.3.2.3 Tool qualification strategy shall consider

a) analysis of tool development process and vendor tool history;

b) adequacy of tool documentation to allow verification of tool output and ease of learning;

c) testing or validation of the tool;

d) evaluation of the tool over a period of use;

e) feedback of experience with tool use.

NOTE Subclause 4.3 contains qualification requirements for the use of pre-developed software that should also be considered for tool qualification strategy.

4.2.3.2.4 Tool outputs should be systematically verified (for example, by test, analysis, or comparison with the output of functionally similar tools), if the output is to be included in the final software.

4.2.3.2.5 If the tool output can introduce faults into the final software and the tool output is not systematically verified, and if there is no mitigation of tool faults (by process diversity or system design), then verification and assessment of the software tool shall be performed as described in 4.3 or the tool shall be developed according to IEC 60880. The qualification process may take into account experience of prior use of the tools where the tool has previously been justified for use in similar applications of the same category where there are similar consequences of failure.

4.2.3.3 Gestion de la configuration des outils

4.2.3.3.1 Tous les outils doivent être sous gestion de la configuration pour assurer l'identification complète des outils sélectionnés (y compris le nom, la version, la variante et éventuellement la configuration) et les paramètres des outils utilisés pour générer les logiciels de référence.

NOTE Cela est utile non seulement pour la cohérence finale du logiciel, mais aussi aide à évaluer l'origine d'un défaut, qui peut se trouver dans le code source, dans l'outil ou dans les paramètres de celui-ci. Il peut aussi être nécessaire dans l'estimation du risque de CCF dû aux outils logiciels.

4.2.3.3.2 Des enregistrements documentant l'historique des erreurs et les limitations des outils doivent être conservés pendant toute la vie du logiciel pour les outils dont la sortie peut injecter directement, ou induire, un défaut dans le logiciel final.

4.2.3.3.3 Toute modification d'un outil doit être vérifiée et évaluée.

4.2.3.4 Traducteurs/compilateurs

Le présent paragraphe présente les prescriptions liées spécifiquement aux traducteurs/compilateurs. Du fait de la taille et de la complexité de nombreux compilateurs, il peut être extrêmement difficile de démontrer qu'un compilateur fonctionne correctement. Toutefois, une grande expérience d'utilisation peut augmenter l'assurance de bon fonctionnement d'un compilateur.

4.2.3.4.1 Il convient de sélectionner les traducteurs et compilateurs sur la base des critères indicatifs concernant les traducteurs/compilateurs donnés dans le présent paragraphe. (Ces critères viennent compléter les prescriptions énoncées dans la CEI 60880, annexe D.)

4.2.3.4.2 Les traducteurs/compilateurs ne doivent pas retirer sans préavis les moyens de programmation défensive ou de contrôle d'erreurs introduits par le programmeur.

4.2.3.4.3 Il convient d'éviter l'utilisation de l'optimisation par compilateur. Celle-ci ne doit pas être utilisée si elle produit du code objet trop difficile à comprendre, à déboguer, à tester et à valider.

NOTE L'optimisation du code peut être utilisée pour satisfaire aux exigences de performances dues à des contraintes liées à la vitesse du matériel ou aux limites de mémoire. Dans des cas exceptionnels, l'utilisation de code assembleur peut être envisagée en plus du remplacement de la plate-forme matérielle.

4.2.3.4.4 Lorsque l'optimisation est utilisée, les tests, la vérification et/ou la validation doivent être effectués sur le code optimisé.

4.2.3.4.5 Les bibliothèques utilisées dans le système cible doivent être considérées comme des ensembles de composants logiciels prédéveloppés. Les composants de la bibliothèque utilisés doivent être évalués, qualifiés et utilisés conformément aux prescriptions de 4.3 relatives à la qualification de logiciels prédéveloppés.

4.2.3.4.6 Des tests, vérifications et/ou validations doivent être effectués afin de s'assurer que le code supplémentaire (instructions d'assembleur) introduit par le traducteur et non directement traçable aux énoncés des lignes sources (code de contrôle d'erreurs, code de traitement d'erreurs et d'exceptions, code d'initialisation, par exemple) est correct.

4.2.3.5 Outils pour les données d'application

Les systèmes informatiques de sûreté nécessitent en général des données applicatives pour définir les signaux, les adresses et les paramètres des fonctions d'application et de service. Les données peuvent être importantes et se composent en général d'informations telles que:

4.2.3.3 Tool configuration management

4.2.3.3.1 All tools shall be under configuration management to ensure the complete identification of selected tools (including name, version, variant, and possibly configuration) and the tool parameters used to generate baselined software.

NOTE This is useful not only for the final software consistency. It also helps in assessing the origin of a fault, which may lie in the source code, in the tool or in the tool parameters. It may also be necessary in the assessment of the potential for CCF due to software tools.

4.2.3.3.2 Records documenting the error history and limitations of tools shall be maintained throughout the life of any tool whose output can introduce a fault into the final software.

4.2.3.3.3 Any modification of a tool shall be verified and assessed.

4.2.3.4 Translators/compilers

This subclause presents requirements specifically related to translators/compilers. The size and complexity of many compilers can make it extremely difficult to demonstrate that a compiler works correctly. However, extensive experience of use can increase confidence that the compiler works correctly.

4.2.3.4.1 Translators/compilers should be selected on the basis of guidance criteria relevant to translators/compilers in this subclause (which complement the requirements in IEC 60880 appendix D).

4.2.3.4.2 Translators/compilers shall not remove without warning defensive programming or error-checking features introduced by the programmer.

4.2.3.4.3 The use of compiler optimization should be avoided. It shall not be used if it produces object code that is excessively difficult to understand, debug, test and validate.

NOTE Code optimization can be used to meet performance requirements due to constraints in hardware speed and storage limits. In exceptional cases, the alternative use of assembly code can be considered in addition to changing the hardware platform.

4.2.3.4.4 Where optimization is used, tests, verification and/or validation shall be performed on the optimized code.

4.2.3.4.5 Libraries which are used in the target system shall be considered as sets of pre-developed software components. Those components of the library used shall be evaluated, qualified and used in accordance with the requirements in 4.3 on qualification of pre-developed software.

4.2.3.4.6 Tests, verification and/or validation shall be carried out to ensure that additional code (assembly instructions) introduced by the translator which is not directly traceable to source line statements (for example, error-checking code, error and exception handling code, initialization code) is correct.

4.2.3.5 Application data tools

Computer-based safety systems usually require application data to define signals, addresses and function parameters of the application functions and service functions. The data can be extensive and normally consists of information such as:

a) références des étiquettes des signaux, description des signaux, implantation des sources et numéros des câbles, types de mesures, plages ou états électriques, unités techniques, définitions des états d'alarme, niveaux d'alarme et de déclenchement;

b) les points de raccordement des signaux, les adresses et pointeurs de bases de données, les adresses et pointeurs d'informations, les adresses et caractéristiques matérielles, le format des affichages, les informations liées aux symboles et couleurs d'affichage, l'identification du contenu des signaux d'affichage, les formats des journaux et des messages internes et le détail de leur contenu;

c) les codes d'action de protection, la priorité ou la logique des alarmes, les signaux d'actionnement, l'identification des opérations logiques et temporisateurs, les états de sortie à adopter en cas de défaillance.

Les données peuvent être extraites de plans, calendriers de conception et des spécifications d'exploitation de la centrale et d'instrumentation des processus. Elles seront traduites pour chargement dans les processeurs cibles du système, puis utilisées pour commander l'action du logiciel en ligne.

Les prescriptions liées à la préparation, la vérification, la validation et la gestion des données pour utilisation en ligne sont présentées ci-dessous.

4.2.3.5.1 La conception des données d'application du système à partir des données d'application de la centrale doit être définie et documentée.

4.2.3.5.2 Les paramètres d'application qui peuvent être modifiés par les opérateurs pendant l'exploitation ainsi que les méthodes pour contrôler ces modifications doivent être identifiés.

4.2.3.5.3 Il convient que les modifications de données d'application ne corrompent ni les autres données, ni le code exécutable du système.

4.2.3.5.4 Le formalisme des procédures pour la vérification et la validation des données doit être semblable aux procédures de vérification et de validation du logiciel, y compris pour ce qui concerne l'identification et l'élimination des erreurs. Des contrôles d'un bout à l'autre doivent être réalisés et doivent inclure chaque étape de la transformation des données, de l'extraction de données sur la base d'informations sur la conception de la centrale jusqu'à l'intégration des structures de données dans le logiciel en ligne, y compris l'utilisation des moyens de transfert des données.

4.2.3.5.5 Il convient que les données à charger dans le logiciel en ligne soient dans une forme adaptée pour pouvoir être imprimées et vérifiées. Autrement un outil doit être utilisé pour reprendre les données et les remettre dans une forme compréhensible pour la vérification.

4.2.3.5.6 Des moyens doivent exister pour permettre la vérification de toutes les configurations de données chargées sur site.

4.2.3.5.7 Si des données définissent l'interface entre deux systèmes, il convient que les données pour chaque système soient générées automatiquement de la même base de données (voir 5.3.1.4 de la CEI 61513).

4.2.3.5.8 Dans certains cas, lorsque la fonctionnalité du logiciel comprenant le traitement, le flux des données et les liaisons d'entrée et sortie est contrôlée ou modifiée par des données de configuration, un justificatif spécifique est requis pour confirmer que les données ont fait l'objet d'un niveau adéquat d'évaluation et successivement de tests. Il peut s'avérer nécessaire d'effectuer à nouveau un volume important de test en cas de changement de telles données.

a) signal tag references, signal descriptions, source locations and cable numbers, measurement types, electrical ranges or states, engineering units, alarm state definitions, alarm and trip levels;

b) signal termination points, data base addresses and pointers, information addresses and pointers, hardware addresses and characteristics, display layouts, display symbol and colour information, display signal content identification, log and internal message formats and details of contents;

c) protection action codes, alarm priority or logic, outputs for action, identification of logic operations and timers, output states to be adopted at failure.

The data may be taken from design drawings, lists and specifications of plant operations and process instrumentation. It will be translated for loading to target processors of the system, and then used to control the action of the on-line software.

Requirements related to the preparation, verification and validation, and management of data for on-line use are presented below.

4.2.3.5.1 The design of the software system application data from the plant application data shall be defined and documented.

4.2.3.5.2 Application parameters that can be changed during operation by the operator shall be identified, together with the methods to be used to control changes of such parameters.

4.2.3.5.3 Changes made to modifiable application data should not corrupt other data and code on the runtime system.

4.2.3.5.4 The formality of procedures for data verification and validation shall be similar to the formality of procedures for software verification and validation, including identification and clearance of errors. End-to-end verification checks shall be performed, and these shall include each stage of data transformation starting from data extraction from plant design information through to incorporation of data structures in the on-line software, including the use of transfer media.

4.2.3.5.5 The data to be loaded to the on-line software should be in a form, which can be printed and verified, or a tool shall be used to take that data and restore it to an intelligible form for verification.

4.2.3.5.6 A facility shall be provided to allow verification of all loaded configuration data on site.

4.2.3.5.7 Where data define the interface between two systems, then the data provided for each system should be automatically generated from the same data base (see IEC 61513, 5.3.1.4).

4.2.3.5.8 In some cases, where software functionality including processes, data flow, and input and output connections are controlled or modified by configuration data, specific documented justification is required to confirm an adequate level of assessment and subsequent testing has been applied to the data. Extensive system retesting may be required following changes to such data.

4.2.3.6 Automatisation des tests

L'automatisation augmente le volume de tests qui peuvent être effectués dans une période déterminée. Ceci peut être obtenu lorsque les exigences suivantes sont remplies

4.2.3.6.1 Il convient que les outils automatisant la validation qui génèrent des données d'essais, qui transportent ou transforment des données et des résultats d'essais et qui évaluent les résultats d'essais, enregistrent un journal complet d'essai. Ceci s'applique au test de modules aussi bien qu'aux simulations de la centrale.

4.2.3.6.2 Il convient que des outils appropriés soient utilisés pour tester et/ou simuler le comportement du code chargé dans le système cible.

4.2.3.6.3 Des outils appropriés doivent être utilisés pour assurer ou vérifier que le bon code exécutable est chargé correctement dans le système.

4.2.3.6.4 Il convient de prendre en considération l'utilisation des outils supplémentaires suivants:

a) générateurs de test, analyseurs de la couverture des tests et bancs de test;

b) programmes de diagnostic en ligne avec des moyens de traçabilité et d'inspection;

c) débogueurs avec des moyens pour déboguer au niveau du code source; et

d) suites de tests automatiques pour faciliter les tests de régression.

4.3 Qualification de logiciels prédéveloppés

4.3.1 Introduction

Le présent paragraphe présente les prescriptions applicables à l'utilisation de logiciels pré-développés (LPD) dans les systèmes de contrôle commande informatisés. Ces prescriptions sont établies comme part des prescriptions de qualification du système dans lequel le LPD est intégré (voir 6.4 de la CEI 61513).

Les LPD pour les systèmes d'I&C peuvent comprendre des petits composants logiciels (par exemple un module d'une bibliothèque de fonctions d'application) ainsi que des gros produits logiciels complexes (par exemple une partie d'un système d'exploitation ou de communication). Les LPD peuvent être classés en deux catégories, selon le type de matériel:

a) LPD polyvalents non spécifiquement développés pour un environnement matériel spécifique, et

b) LPD intégrés dans des composants du matériel, qui sont à utiliser en association avec ces composants.

Des composants LPD son dits réutilisables quand ils peuvent être utilisés dans différents programmes informatiques ou systèmes, par exemple comme partie d'une famille d'équipements (plate-forme d'équipements). Il est possible que des composants, qui sont indépendants des particularités propres aux applications de la centrale, aient déjà été qualifiés pour l'utilisation dans des systèmes qui réalisent des fonctions de catégorie A.

4.3.1.1 Prescriptions de la CEI 60880 liées à l'utilisation des LPD

La CEI 60880 examine l'utilisation des LPD et les prescriptions de cette norme sont résumées à l'article D.1 de la présente norme; ils s'appliquent en complément des prescriptions établies ici.

4.2.3.6 Automation of testing

Automation increases the amount of testing that can be performed in a given period. This can be achieved by meeting the following requirements.

4.2.3.6.1 Automated validation tools that generate test data, transport or transform test data and test results and evaluate test results should record a complete test log. This is applicable to module tests as well as to plant simulations.

4.2.3.6.2 Appropriate tools should be used to test and/or simulate the behaviour of the executable code loaded in the target system.

4.2.3.6.3 Appropriate tools shall be used to ensure or verify that the right executable code is loaded correctly in the target system.

4.2.3.6.4 The use of the following additional tools should be considered:

a) test generators, test coverage analysers and test drivers;
b) on-line diagnostic programs with dump inspect and trace facilities;
c) debuggers with debugging facilities at the source code level; and
d) automated test suites to facilitate regression testing.

4.3 Qualification of pre-developed software

4.3.1 Introduction

This subclause gives requirements for the use of pre-developed software (PDS) in I&C computer-based systems. These requirements are established as part of the qualification requirements of the system in which the PDS is integrated (see 6.4 of IEC 61513).

PDS for I&C systems may range from small software components (for example, an application function library module), to large and complex software products (for example, parts of an operating system, or communication drivers). PDS may be divided into two types with respect to hardware:

a) general-purpose PDS that has not been specifically developed for a specific hardware environment; and
b) PDS integrated in hardware components that has to be used in association with this hardware.

PDS components are called reusable when they can be used in different computer programs or systems, for example, as part of an equipment family (equipment platform). Those components which are independent of specific plant application details may have already been qualified for use in systems performing category A FSE.

4.3.1.1 IEC 60880 requirements for the use of PDS

IEC 60880 contemplates the use of PDS and the requirements of that standard are summarized in clause D.1 of this standard; these requirements apply in addition to the requirements stated here.

4.3.1.2 Principes directeurs de l'utilisation des LPD

Les spécifications de nouveaux systèmes de sûreté tendent souvent à utiliser des équipements prédéveloppés pour réaliser une partie ou la totalité d'un "système nouveau" (voir 6.1.2.1 de la CEI 61513). L'utilisation des équipements prédéveloppés peut être intéressante pour la productivité et la fiabilité du système lorsque ces équipements sont introduits de manière appropriée et que leur qualité est correcte. Lorsque des LPD ont été utilisés dans de nombreuses applications similaires, les gains résultant de cette expérience en exploitation peuvent être mis en évidence dans le processus de qualification. En particulier, la réutilisation de LPD validés peut accroître la confiance dans la fiabilité du système.

4.3.2 Prescriptions générales

4.3.2.1 Les LPD envisagés pour l'utilisation dans un système doivent être évalués par rapport aux critères développés à partir de la présente norme, et agréés comme appropriés et adaptés à l'utilisation compte tenu de la catégorie de sûreté des fonctions réalisées.

4.3.2.2 Le processus d'évaluation du LPD doit

a) déterminer la capacité des LPD à satisfaire les prescriptions fonctionnelles, de performances et architecturales du cahier des charges du système informatique (voir 6.1.1 de la CEI 61513), et par conséquent leur aptitude à l'usage;

b) déterminer les modifications requises pour corriger ou adapter le LPD;

c) agréer la qualité du LPD; et

d) évaluer l'expérience acquise en exploitation du LPD, quand cela est nécessaire pour les évaluations mentionnées ci-dessus.

4.3.2.3 Les conclusions du processus d'évaluation doivent être documentées.

NOTE Dans la présente norme, le terme agrément est utilisé pour décrire une action conduite par l'organisation responsable du développement du système informatique (ou au nom de cette organisation). Il n'est ni impliqué, ni exigé, que cet agrément soit conduit par l'organisme de sûreté, mais celui-ci peut toutefois décider de le faire.

4.3.3 Processus d'évaluation et d'agrément

Le processus d'évaluation et agrément du LPD doit comprendre:

a) une évaluation des caractéristiques fonctionnelles et des performances du LPD et de la documentation existante de qualification (voir 4.3.3.1);

NOTE Pour les LPD intégrés dans un produit, ces caractéristiques peuvent être exprimées comme des propriétés du produit en accord avec la CEI 61069-2.

b) une évaluation de la qualité de la conception et du développement du logiciel (voir 4.3.3.2);

NOTE Dans le cas des logiciels validés réutilisables, seule l'évaluation de l'appropriation à l'usage est nécessaire, l'évaluation de la qualité étant impliquée par sa validation.

c) une évaluation de l'expérience acquise en exploitation lorsqu'elle est nécessaire pour compenser des faiblesses dans la démonstration obtenue par a) et b), (voir 4.3.3.3); et

d) un agrément détaillé et documenté de la qualité des preuves obtenues à la suite de l'évaluation détaillée et des travaux supplémentaires associés, qui permettra de déclarer le LPD bon pour utilisation dans le système.

La figure 1 montre les relations entre les différentes étapes du processus d'évaluation et agrément du LPD.

La figure 2 montre les relations entre ce processus et la qualification du système.

NOTE Le processus décrit dans le présent paragraphe est une vision simplifiée de la réalité et, en tant que tel, ne fait pas état de toutes les itérations ou chevauchements entre activités d'évaluation ainsi qu'entre ces processus et les activités de spécification et de développement du système informatique.

4.3.1.2 Rationale for the use of PDS

The specifications of new safety systems often identify the use of pre-developed equipment including PDS to implement part or the whole of a "new system" (see 6.1.2.1 of IEC 61513). Use of pre-developed equipment can be beneficial to productivity and the reliability of the system when these items are of suitable quality and introduced in a proper manner. When PDS items have been used in many applications similar to the intended use, a benefit from this operating experience can be claimed in their evaluation. In particular, the reuse of validated PDS can increase confidence in the reliability of the system.

4.3.2 General requirements

4.3.2.1 A PDS that is a candidate for use as part of a system shall be evaluated and assessed against criteria developed from this standard, as being appropriate and suitable for use given the safety category of the implemented functions.

4.3.2.2 The evaluation of the PDS shall

a) determine the capability of the PDS to meet the functional, performance and architectural requirements of the system requirements specification (see 6.1.1 of IEC 61513), and its resulting suitability;

b) identify any modifications needed to correct or adapt the PDS;

c) assess the quality of the PDS; and

d) evaluate the operating experience of the PDS, when required for the above evaluations.

4.3.2.3 The conclusions of this evaluation process shall be documented.

NOTE In this standard, assessment is used to describe an action made by the organization in charge of the computer-based system development (or on behalf of this organization); it is neither implied nor required that assessment be made by licensing authorities, although they may also choose to do so.

4.3.3 Evaluation and assessment process

The PDS evaluation and assessment process shall include

a) an evaluation of the functional and performance features of the PDS and existing qualification documentation (see 4.3.3.1);

NOTE For PDS integrated in a product these features may be expressed as product properties according to IEC 61069-2.

b) a quality evaluation of the software design and development process (see 4.3.3.2);

NOTE For reusable pre-assessed software, only the evaluation of suitability needs to be made; the evaluation of quality is implied by its validation.

c) an evaluation of operating experience if needed to compensate for weaknesses in demonstration gained from both a) and b) (see 4.3.3.3); and

d) a comprehensive documented assessment of the evidence from the above evaluations, and associated complementary work, which will enable the PDS to be accepted for use in the system.

Figure 1 shows the relations between the different stages of the evaluation and assessment process of the PDS.

Figure 2 shows the relationship between this process and the qualification of the system.

NOTE The process described in this subclause is a simplified view of reality and, as such, does not show all the iterations or overlap between the evaluation activities as well as between these processes and the computer-based system specification and development activities.

4.3.3.1 Evaluation de l'aptitude à l'usage

L'objectif de ce processus est de confirmer que les spécifications fonctionnelles, de performances et architecturales du LPD sont en accord avec les exigences pour le LPD spécifiées dans le cahier de charges. Le processus identifie les composants directement aptes à l'usage dans le système de la centrale et aussi les composants qui nécessitent des modifications.

NOTE L'évaluation est basée sur l'analyse des spécifications et de la documentation fonctionnelle.

Il convient que l'évaluation de l'aptitude à l'usage soit initiée dans un stade précoce de la spécification du système (voir 6.2 de la CEI 61513). Cette évaluation doit être complétée afin:

- d'aider les concepteurs dans la sélection de la conception architecturale du système;
- d'obtenir confirmation vérifiable que les spécifications fonctionnelles et de performances du LPD sont conformes aux spécifications des prescriptions du système.

4.3.3.1.1 Documentation d'entrée requise

4.3.3.1.1.1 La documentation suivante doit être disponible:

- la documentation de spécification du système, qui définit les exigences fonctionnelles, d'interface et de performances qui doivent être remplies par le LPD dans le cadre de l'architecture du système (voir 6.1.1 et 6.1.2 de CEI 61513);
- les documents de spécification et les documents utilisateur. Il convient que ces documents définissent explicitement toutes les caractéristiques liées au respect des spécifications fonctionnelles et de performances du système. Des analyses ou tests doivent être réalisés afin de rendre explicites les caractéristiques mises en œuvre si elles ne sont pas explicitement définies.

4.3.3.1.1.2 Le LPD doit être sous contrôle de configuration; sa version et sa configuration doivent être connues de façon précise.

4.3.3.1.2 Exigences pour l'évaluation de l'aptitude à l'usage

Extension de l'exigence de 4.3.2.2.

4.3.3.1.2.1 Les spécifications du LPD doivent être évaluées par rapport aux spécifications des exigences du système (voir 6.1.1 de la CEI 61513). Si des non-conformités sont identifiées, le LPD doit être refusé ou modifié, ou les spécifications de prescriptions doivent être adaptées pour résoudre les non-conformités, sous réserve que la fonction globale de sûreté soit préservée.

4.3.3.1.2.2 S'il est nécessaire de modifier le LPD, il doit être procédé à une évaluation, sur la base de la documentation relative à la conception du système, afin de déterminer si ces modifications sont réalisables en accord avec la CEI 60880. Si la modification ne peut pas être réalisée de manière conforme, le LPD doit être refusé.

NOTE L'évaluation de la qualité indique si ces modifications sont faisables (voir 4.3.3.2.2). La réalisation correspondante est effectué dans le cadre du cycle de vie du système (voir l'article 6 de la CEI 61513).

4.3.3.1.2.3 Dans le cas des LPD contenus dans une bibliothèque, sauf dans le cas où la totalité de la bibliothèque a été agréée, il devrait être possible de découper la bibliothèque pour réaliser une bibliothèque restreinte correspondante aux exigences du logiciel et de lier le programme avec les modules de cette bibliothèque réduite, qui doit être constituée de modules validés.

4.3.3.1 Evaluation of suitability

The objective of this process is to confirm that the functional, performance and architectural specifications of the PDS comply with the requirements for the PDS specified in the system specification. This process identifies components directly suitable for use in the plant system and also identifies those that will require modification.

NOTE The evaluation is based on the analysis of the specifications and functional documentation.

The evaluation of the suitability of the PDS should be initiated in an early stage of the system specification (see 6.2 of IEC 61513) and it shall be completed in order to

- assist the designers in the architectural design of the system;
- gain auditable evidence that the functionality and performance of the PDS meets the requirements of the system.

4.3.3.1.1 Required input documentation

4.3.3.1.1.1 The following documentation shall be available:

- system specification documentation which identifies the functional, interface and performance requirements, to be fulfilled by the PDS in the framework of the system architecture (see 6.1.1 and 6.1.2 of IEC 61513);
- the PDS specification and user documentation. These documents should explicitly define all the characteristics that are relevant in fulfilling the system functional and performance specifications. Analysis or test shall be performed to make the implemented characteristics explicit if they are not explicitly defined.

4.3.3.1.1.2 The PDS shall be under configuration management; the version and configuration of the PDS shall be known precisely.

4.3.3.1.2 Evaluation requirements for suitability

Expanding on requirement of 4.3.2.2:

4.3.3.1.2.1 The specifications of the PDS shall be evaluated with respect to the system requirements specification (see 6.1.1 of IEC 61513). If discrepancies exist, the PDS shall either be rejected or modified or the requirement specifications shall be adapted to resolve them, provided that the overall safety function is preserved.

4.3.3.1.2.2 If it is necessary to modify the PDS an evaluation shall be completed, based on the PDS design documentation, to determine if the change can be performed in a manner compliant with IEC 60880. If the change cannot be performed in a compliant manner, the use of the PDS shall be rejected.

NOTE The quality evaluation indicates if these modifications are feasible (see 4.3.3.2.2). The corresponding implementation is handled in the frame of the system life cycle (see clause 6 of IEC 61513).

4.3.3.1.2.3 For a PDS that is contained in a library, except when the whole library has to be assessed, it should be possible to tailor the library to build a restricted library meeting the software needs and to link the program with this restricted library which shall be composed of assessed components.

4.3.3.1.2.4 L'évaluation d'aptitude à l'usage doit identifier les fonctions incluses dans le LPD qui ne sont ni voulues ni nécessaires dans le système et les mesures pour s'assurer que ces fonctions ne vont pas interférer avec les fonctions de sûreté.

4.3.3.1.2.5 Lorsque l'évaluation a été effectuée, un document doit être produit pour consigner

a) que les spécifications fonctionnelles et de performances du LPD sont conformes aux spécifications des prescriptions du système; ou

b) que le LPD doit être rejeté du fait qu'il n'est pas approprié à l'usage.

4.3.3.2 Evaluation de la qualité

L'objectif de cette évaluation est de démontrer que les caractéristiques du LPD sont en accord avec les exigences pour un système qui réalise des fonctions de catégorie A et qu'une AQ adéquate a été exercée le long du cycle de vie du LPD. Cette évaluation est basée sur la documentation de la conception et du plan qualité logiciel du LPD mais peut nécessiter l'analyse de l'expérience acquise en exploitation.

NOTE A l'opposé de l'évaluation d'aptitude à l'usage, qui est essentiellement une approche «boîte noire», l'évaluation de la qualité nécessite des analyses «boîte claire».

4.3.3.2.1 Documentation d'entrée

4.3.3.2.1.1 En supplément de ce qui est requis en 4.3.3.1.1, la documentation suivante doit être disponible:

- la documentation de spécification du système, qui définit l'importance pour la sûreté des fonctions remplies par le LPD dans le cadre de l'architecture du système (voir 6.1.2 de la CEI 61513 et annexe A de la CEI 60880);

NOTE Dans l'évaluation de la qualité, le niveau de confiance à atteindre sur le fait que le LPD fonctionnera en accord avec les spécifications sera différent pour les trois catégories, le niveau de confiance le plus élevé étant requis pour la catégorie A (voir 8.2.1 de la CEI 61226).

- Les documents de qualification du LPD, y compris les certifications précédentes ou les agréments indépendants, si ceux-ci sont utilisés pour l'agrément.

4.3.3.2.1.2 La documentation suivante relative au LPD doit être fournie ou l'utilisation d'informations alternatives doit être justifiée:

NOTE 1 L'étendue de la documentation requise pour le LPD dépend de différents facteurs (voir l'article D.2)

- le plan qualité logiciel (division en tâches élémentaires et activités associées) utilisé dans le cycle de vie logiciel du LPD (voir article 3 de la CEI 60880) et les tâches et procédures d'assurance de la qualité correspondantes (en particulier le plan de vérification);
- les documents de spécification, de développement (conception et codage) et de maintenance et les documents de vérification correspondants;
- le plan d'intégration logiciel/matériel et la vérification associée;
- la validation et les essais effectués sur le LPD par le fournisseur ou le client.

NOTE 2 Pour les deux derniers points: cette documentation est nécessaire seulement si le LPD est intégré dans des composants matériels.

4.3.3.2.1.3 La documentation de l'expérience acquise en exploitation du LPD doit être disponible si elle est utilisée dans l'évaluation pour compenser des manques dans la documentation ci-dessus ou pour justifier l'utilisation de pratiques différentes de celles de la CEI 60880 et de la présente norme.

4.3.3.1.2.4 The suitability evaluation shall identify the functions that are included in the PDS which are unintended and unneeded by the system and also the measures to ensure that these functions do not interfere with safety functions.

4.3.3.1.2.5 When the evaluation is concluded, a document shall be produced to record

a) whether the functional and performance specifications of the PDS comply with the software requirement specifications of the system; and

b) where the PDS is not adequate, the grounds for rejection.

4.3.3.2 Quality evaluation

The objective of this evaluation is to provide evidence that the features of the PDS design are appropriate for a system performing a category A FSE, and that adequate QA has been exercised through the life cycle of the PDS. This evaluation is based on the design and software quality plan documentation of the PDS but may also require analysis of its operating history.

NOTE As opposed to the suitability evaluation, which is essentially a black-box approach, the quality evaluation requires white-box analysis.

4.3.3.2.1 Input documentation

4.3.3.2.1.1 The following documentation shall be available, in addition to that required in 4.3.3.1.1:

- the system specification documentation which identifies the importance to safety of the functions implemented with the PDS in the architectural design of the system (see 6.1.2 of IEC 61513 and appendix A of IEC 60880);

NOTE The level of assurance to be achieved by the quality evaluation that the PDS will perform as specified will be different for the three categories, with category A requiring the highest assurance (see 8.2.1 of IEC 61226).

- the qualification documentation of the PDS, including the documentation on previous certifications or independent assessments of the PDS, if it is to be used for the assessment.

4.3.3.2.1.2 The following documentation related to the PDS shall be provided or the use of alternative information shall be justified:

NOTE 1 The completeness of the documentation required for the PDS depends on different factors (see clause D.2).

- the software quality plan (division in elementary tasks and associated activities) used in the software life cycle of the PDS (see clause 3 of IEC 60880), and the corresponding quality assurance tasks and procedures records (notably verification planning);
- the specification, development (design and coding) and maintenance documents, and the corresponding verification documents;
- the software/hardware integration plan and associated verification;
- the validation plan and tests performed on the product by the vendor or customer.

NOTE 2 For the latter two points, this documentation is needed only if the PDS is integrated in hardware components.

4.3.3.2.1.3 Documentation of operating experience of the PDS shall be available if it is to be used in the evaluation to compensate for lack of the above documentation or to justify use of practices differing from those of IEC 60880 and this standard.

4.3.3.2.1.4 Il convient que la documentation donne des informations sur des facteurs industriels tels que la distribution du LPD et l'assistance clients.

4.3.3.2.2 Exigences de l'évaluation de la qualité

4.3.3.2.2.1 Les exigences du plan qualité logiciel du LPD et la vérification et documentation correspondantes doivent être évaluées par rapport aux prescriptions de la CEI 60880. Cette analyse de conformité nécessite des interprétations pour identifier les exigences qui sont applicables dans le contexte d'utilisation du LPD dans le système (voir l'article D.3).

4.3.3.2.2.2 La conception du LPD doit être en accord avec les contraintes sur l'architecture et le comportement déterministe du système.

4.3.3.2.2.3 Lorsque des procédures différentes de celles des annexes A à F de la CEI 60880 ont été utilisées pour le développement du LPD, leur adéquation doit être analysée et justifiée en accord avec l'article 1 de la CEI 60880. Leur importance dans l'assurance des caractéristiques de qualité logicielle doit être évalué conjointement avec les prescriptions du système. Les résultats de l'évaluation et de l'analyse doivent être consignés pour un examen indépendant.

4.3.3.2.2.4 Les non-conformités par rapport aux exigences de la CEI 60880, les propriétés qui ne peuvent pas être vérifiées et les faiblesses ou les étapes manquantes dans le processus de vérification ou de documentation doivent être identifiées. Chacune doit être graduée selon son importance pour l'assurance des caractéristiques de qualité logicielle et l'importance pour la sûreté des fonctions remplies dans le système. L'article D.4 donne des directives pour la graduation des non-conformités.

4.3.3.2.2.5 Si un système qui réalise des fonctions de catégorie A comprend aussi des fonctions de catégorie inférieure devant être réalisées par un LPD, et si la conception architecturale du système est telle que ce LPD peut mettre en danger les fonctions de catégorie A (voir 6.2 de la CEI 61513), alors les critères d'évaluation pour la mise en œuvre de fonctions de catégorie A doivent être appliqués à ce LPD.

4.3.3.2.2.6 La documentation de qualification doit démontrer que le LPD intégré dans le matériel a été validé pour démontrer qu'il remplit ses spécifications fonctionnelles et de performances.

NOTE Le comportement fonctionnel et la performance des composants LPD peuvent être qualifiés implicitement lors de la qualification fonctionnelle des équipements individuels dans lesquels ils sont intégrés (voir 2 de la figure 2). Toutefois, il y a des propriétés qui peuvent être qualifiées seulement en utilisant des configurations d'équipements.

4.3.3.2.2.7 Lorsque des composants du LPD ont des caractéristiques qui ne peuvent être validées que dans le cadre d'une configuration du système, alors la validation de ces caractéristiques doit être réalisée sur la configuration finale du système.

4.3.3.2.2.8 La qualité et le niveau de couverture des tests de validation effectués sur le LPD doivent être évalués par rapport aux prescriptions des articles 7 et 8 de la CEI 60880 et des tests de validation supplémentaires doivent être réalisés si nécessaire.

4.3.3.2.2.9 Lorsque l'évaluation de la conception et du cycle de vie est terminée, un document doit être produit pour consigner que

a) la qualité du LPD a été démontrée et aucun test supplémentaire ni aucune analyse de l'expérience acquise en exploitation n'est requise;

b) une qualification complémentaire doit être effectuée lorsque le système configuré sera disponible;

4.3.3.2.1.4 The documentation should provide information on industrial factors such as the distribution of the PDS and the customer support.

4.3.3.2.2 Evaluation requirements for quality

4.3.3.2.2.1 The requirements of the PDS software quality plan and the corresponding verification and documentation shall be evaluated for conformance with the requirements of IEC 60880. This analysis of conformance needs interpretation to identify the requirements which are applicable in the context of the use of the PDS in the system (see clause D.3).

4.3.3.2.2.2 The PDS design shall be consistent with the constraints on the architecture and deterministic internal behaviour of the system.

4.3.3.2.2.3 If practices differing from those of appendices A to F of IEC 60880 have been used for the development of the PDS, their adequacy shall be analysed and justified according to clause 1 of IEC 60880. Their importance in the assurance of the software quality characteristics shall be evaluated in conjunction with the system requirements. The results of the evaluation and analysis shall be recorded for independent review.

4.3.3.2.2.4 Non-conformities to IEC 60880 requirements, properties that cannot be verified, weakness or missing steps in the verification or documentation process shall be identified. Each shall be ranked according to its importance in the assurance of the software quality characteristics, and the importance to safety of the functions implemented in the system. Clause D.4 provides guidance for the ranking of non-conformities.

4.3.3.2.2.5 If systems implementing category A functions include functions of a lower category that are to be performed by a PDS, and the system architectural design is such that this PDS could potentially jeopardize the category A functions (see 6.2 of IEC 61513), then the evaluation criteria for software implementing category A functions shall be applied to such PDS.

4.3.3.2.2.6 The qualification documentation shall provide evidence that PDS integrated in hardware components, has been validated to demonstrate that it meets its functional and performance specifications.

NOTE The functional and performance behaviour of the PDS components may be implicitly qualified by the functional qualification of the individual equipment in which they are integrated (see 2 of figure 2). However, there are properties that may only be qualified using configurations of equipment.

4.3.3.2.2.7 Where PDS components contain features that cannot be validated other than in the final system configuration, then validation of these features shall be performed in the final system configuration.

4.3.3.2.2.8 The quality and degree of coverage of the validation tests performed on the PDS shall be evaluated with reference to the requirements of clauses 7 and 8 of IEC 60880 and additional validation tests performed if necessary.

4.3.3.2.2.9 When the evaluation of the design and of the life cycle is concluded, a document shall be produced to record that

a) the PDS quality has been proved and no additional tests or analysis of operating experience is required;

b) complementary qualification shall be performed when the system configuration is available;

c) des faiblesses et un manque d'information ont été détectés dans les thèmes soumis à évaluation, mais ils peuvent être compensés par des documentations, des vérifications, des analyses et/ou des tests de validation supplémentaires;

d) des faiblesses et un manque d'information ont été détectés dans l'évaluation, mais ils peuvent être compensés sur la base de l'expérience acquise en exploitation;

e) le LPD (ou des parties du LPD) nécessite des modifications pour son utilisation prévue dans le système (voir 4.3.3.3) et il a le niveau de qualité approprié; les modifications peuvent être effectuées en accord avec la CEI 60880;

NOTE Après réalisation des modifications définies, une évaluation et un agrément complémentaires sont nécessaires dans le cadre de la qualification du système (voir 5 de la figure 2).

f) il faut s'attendre à des problèmes importants lors du transfert du LPD dans le nouveau matériel;

g) la qualité du LPD n'est pas adéquate et le LPD doit être rejeté pour le motif que les faiblesses sont trop importantes ou que les informations sont trop insuffisantes pour pouvoir être compensées; et

h) l'indépendance des fonctions ou propriétés du LPD qualifiées par rapport à celles non qualifiées a ou n'a pas été établie.

4.3.3.3 Evaluation de l'expérience acquise en exploitation

L'objectif de cette évaluation est de démontrer qu'une expérience appropriée acquise en exploitation du LPD peut compenser des faiblesses et des manques d'information détectés dans l'évaluation de la qualité.

Parmi la totalité des fonctions/propriétés du LPD, les fonctions/propriétés à qualifier sur la base du retour d'expérience doivent être identifiées et ce qui suit doit être évalué pour le LPD:

a) les méthodes pour le recueil des données sur l'expérience en exploitation;

b) les méthodes pour l'enregistrement du temps d'exploitation de la version du LPD et pour produire l'historique d'exploitation;

c) l'historique opérationnel des faits techniques, défauts et comptes rendus d'erreurs; et

d) l'historique opérationnel des modifications suite à des défauts ou pour d'autres raisons.

4.3.3.3.1 Validation des données d'entrée et des méthodes pour produire l'historique d'exploitation

Les données liées à l'évaluation de l'expérience acquise en exploitation du LPD sont obtenues auprès du fournisseur et si possible des utilisateurs de systèmes utilisant le LPD. Pour que l'expérience acquise en exploitation soit considérée apte à l'évaluation du LPD, il faut valider les méthodes utilisées pour recueillir les données et pour produire l'historique d'exploitation.

4.3.3.3.1.1 Seules des informations issues d'un processus de recueil de données bien défini et contrôlé doivent être utilisées.

4.3.3.3.1.2 Les procédures de recueil doivent être évaluées afin de valider l'exhaustivité et la crédibilité des données (l'article D.5 donne des directives pour le recueil et la validation des données).

4.3.3.3.1.3 Seule l'expérience acquise en exploitation qui a été collectée sous des conditions similaires à celles de l'utilisation prévue doit être considérée comme valide.

4.3.3.3.1.4 Le temps d'exploitation cumulé du LPD objet de l'évaluation doit être établi. Il peut être calculé en additionnant le temps d'exploitation de chaque installation pour lequel un retour d'expérience a été recueilli et validé. Il convient d'exclure les temps pendant lesquels le LPD n'a pas été utilisé de manière représentative.

c) lack of information has been detected during the evaluation, but this can be compensated by the completion of additional verification and validation, testing or code analysis and documentation;

d) lack of information has been detected during the evaluation, which can be compensated for by use of operating experience;

e) the PDS (or part of the PDS) requires modification for the intended use in the system (see 4.3.3.3), and that it has the appropriate level of quality so the modifications may be performed in accordance with IEC 60880;

NOTE After satisfactory completion of the defined modifications, a complementary evaluation and assessment is needed in the frame of the system qualification (see 5 of figure 2).

f) significant problems can be expected because of the transfer of the PDS to new hardware;

g) the PDS quality is not adequate and the PDS shall be rejected on grounds that the weaknesses are too great or the information inadequate for effective compensation; and

h) the independence of the qualified functions/properties of the PDS from those not qualified has/has not been established.

4.3.3.3 Evaluation of operating experience

The objective of this evaluation is to provide evidence that suitable operating experience of the PDS may compensate for deficiencies detected in the quality evaluation.

Those functions/properties of the PDS to be evaluated on the basis of feed-back of experience shall be identified and the following shall be evaluated:

a) the methods for collection of data on operating experience;

b) the methods for recording the PDS version operating time and for producing the operating history;

c) the operational history of findings, defects and error reports; and

d) the operational history of modifications made for defects or other reasons.

4.3.3.3.1 Validation of input data and methods for producing the operating history

The evaluation of operating experience of the PDS is based on data available from the vendor and, if possible, from users of systems running the PDS. In order to consider the operating experience suitable for the evaluation of the PDS, the methods for collection of data and producing the operating history have to be validated.

4.3.3.3.1.1 Only information that is derived from a well-defined and controlled data collection process shall be used.

4.3.3.3.1.2 Collection procedures shall be assessed to validate the data for completeness and credibility. Clause D.5 gives guidance for collection and validation of data.

4.3.3.3.1.3 Operating experience shall be considered valid only if it is observed under conditions similar to the conditions during intended operation.

4.3.3.3.1.4 The accumulated operating time for the PDS under evaluation shall be established. It may be calculated by adding together the operating time of each installation for which operational experience has been collected and validated. Time in which the PDS was not operating in a representative manner should be excluded.

4.3.3.3.1.5 Les temps d'exploitation à prendre en compte doivent faire référence à des LPD de la même version que celle dont l'utilisation est prévue. Lorsque les temps d'exploitation d'autres versions sont inclus, une analyse des différences et de l'historique de ces versions doit être effectuée. Cette analyse doit identifier les parties et les fonctions du LPD qui diffèrent, et les parties et fonctions qui ne sont pas affectées par les modifications afin de démontrer que cet historique est valide.

4.3.3.3.1.6 Les problèmes et défaillances survenus ainsi que leurs corrections dans les différentes versions du LPD doivent être analysés et classés en fonction de leur sévérité. Leurs conséquences sur les fonctions désirées doivent être évaluées.

4.3.3.3.1.7 Aucune des fonctions qualifiées ne doit être affectée par les erreurs rencontrées ou les modifications apportées sur d'autres fonctions d'un même LPD.

4.3.3.3.1.8 Il convient que l'évaluation des fonctions de communication prenne en compte l'expérience acquise en exploitation. Il convient que les limites de service soient identifiées et comparées aux prévisions pour les conditions normales, de charge de pointe et de défaillance d'équipements.

4.3.3.3.1.9 Il convient que l'expérience acquise en exploitation satisfasse aux critères d'acceptation suivants:

a) le LPD a accumulé un temps d'exploitation suffisant (voir le point b) de l'article D.5);

NOTE Il convient que le temps d'exploitation suffisant soit déterminé au cas par cas sur la base du jugement technique. Il convient que ce jugement prenne notamment en compte le niveau de fiabilité prévisionnel requis au niveau système pour les fonctions pour lesquelles le LPD est utilisé.

b) aucune modification importante n'a été faite ni aucune erreur importante détectée pendant une période d'exploitation significative sur plusieurs sites ou applications;

c) le LPD a de préférence été exploité sur plusieurs installations.

4.3.3.3.1.10 Un document d'évaluation doit fournir les conclusions de l'évaluation de l'expérience acquise en exploitation et indiquer

a) si l'expérience acquise en exploitation pour les fonctions/propriétés du LPD est appropriée et son utilisation pour soutenir l'évaluation de la qualité du LPD est justifiée; ou

b) si l'expérience acquise en exploitation n'est pas appropriée ou suffisamment éprouvée.

4.3.3.3.2 Critères d'acceptation de l'expérience acquise en exploitation comme facteur de compensation

L'expérience acquise en exploitation peut être utilisée comme un facteur de compensation pour l'acceptation du LPD, si les critères suivants sont remplis.

4.3.3.3.2.1 L'évaluation du retour d'expérience ne doit jamais remplacer l'évaluation de la conception du produit lui-même et de sa documentation (voir 4.3.3.2.2).

4.3.3.3.2.2 L'expérience acquise en exploitation doit être acceptée comme une partie de la justification seulement si elle est utilisée pour compenser des lacunes dans l'évaluation du LPD par rapport aux recommandations sur la conception du point c) de l'article B.2 de la CEI 60880 sur les systèmes d'exploitation et les programmes standards.

4.3.3.3.2.3 L'expérience acquise en exploitation doit être acceptée seulement comme une partie de la justification si elle démontre qu'aucun défaut suspecté ou connu n'est susceptible d'empêcher le fonctionnement d'une fonction de catégorie A ou de la faire fonctionner incorrectement dans le système.

4.3.3.3.1.5 The operating times shall be for the same version of the PDS that is intended to be used. When operating time of other versions is included, an analysis of the differences and history of these versions shall be made. This analysis shall identify the parts and functions of the PDS which differ, and the parts and functions which are not affected by the modifications to demonstrate that the history is valid.

4.3.3.3.1.6 Problems, failures and their correction in the different versions of the PDS shall be analysed and classified according to their severity. Their impact on the intended functions shall be evaluated.

4.3.3.3.1.7 No qualified function shall be affected by errors found or modifications made to any other functions of the same PDS.

4.3.3.3.1.8 Evaluation of communication functions should take account of operating experience. The service limits should be identified and compared with the forecasts for normal, peak load and equipment failure conditions.

4.3.3.3.1.9 Operating experience should be considered as suitable when the following criteria are met:

a) the PDS has achieved a sufficient accumulated operating time (see item b) of clause D.5);

NOTE The sufficient operating time should be determined on a case-by-case decision based on engineering judgement. This judgement should take into account notably the anticipated reliability level required at system level for the functions in which the PDS is used.

b) no significant modifications have been effected and no errors are detected over a significant operating time on several sites or applications; and

c) the PDS has preferably operated on several installations.

4.3.3.3.1.10 When the evaluation of the operating experience is completed, a document shall be produced to record

a) if the operating experience is suitable for the identified functions/properties of the PDS and justify why it may be used to support assessment of the quality; or

b) that the operating experience is not suitable or is not sufficiently proved.

4.3.3.3.2 Acceptance criteria for the operating experience as a compensating factor

Suitable operating experience may be used as a compensating factor for acceptance of the PDS, when the following criteria are met.

4.3.3.3.2.1 The evaluation of the feedback of operating experience shall never replace the evaluation of the design of the product itself and of its documentation (see 4.3.3.2.2).

4.3.3.3.2.2 Suitable operating experience shall be accepted as part of the justification only if it is used to compensate for weaknesses in the assessment of a PDS against the design recommendations of item c) of clause B.2 of IEC 60880 on operating systems and standardized programs.

4.3.3.3.2.3 The operational history shall be accepted as part of the justification only if it demonstrates that no suspected or known defects are able to prevent operation of a category A function or cause it to act incorrectly in the system.

4.3.3.3.2.4 La rigueur de l'analyse du retour d'expérience doit être cohérente avec la fiabilité du système et il convient que les preuves d'absence de défauts techniques fournies par cette analyse soient cohérentes avec celles réalisables lorsque la CEI 60880 est appliquée.

4.3.3.3.2.5 Un document d'évaluation final doit être rédigé et indiquer

a) qu'une expérience d'exploitation satisfaisante compense les faiblesses détectées dans l'évaluation de la conception et du cycle de vie du LPD; ou

b) que l'utilisation du LPD est rejetée parce que les résultats de l'évaluation sont négatifs, ou parce que l'expérience acquise en exploitation n'est pas encore suffisante pour compenser les faiblesses identifiées au cours du développement et de la validation.

4.3.3.4 Justification globale

4.3.3.4.1 Lorsque les évaluations et le travail complémentaire nécessaire (modifications du LPD, tests complémentaires, documentation complémentaire) ont été achevés, un document de justification global pour l'utilisation du LPD dans la mise en oeuvre du système informatique doit être rédigé.

4.3.3.4.2 Le document, basé sur les conclusions des évaluations décrites en 4.3.3.1, 4.3.3.2 et 4.3.3.3, doit enregistrer l'agrément qui démontre que le LPD (ou une partie du LPD) est adapté et a le niveau de qualité approprié pour l'usage prévu dans le système et qu'aucune autre modification n'est nécessaire.

4.3.4 Prescriptions liées à l'intégration dans le système et à la maintenance des LPD

4.3.4.1 Après le processus d'évaluation et d'estimation, la décision d'utiliser le LPD doit être prise formellement et documentée dans le cadre du projet à la suite d'une revue formelle de conception.

4.3.4.2 Lorsqu'un LPD est utilisé pour la mise en oeuvre de tout ou partie d'un système informatique, les procédures d'intégration du LPD doivent être décrites dans le plan d'assurance qualité et dans le plan d'intégration du système informatique (voir 6.2.1 et 6.2.3 de la CEI 61513).

4.3.4.3 Après acceptation, le LPD doit être placé sous la gestion de configuration du système (voir 6.2.1.2 de la CEI 61513) et seules la version soumise à la qualification décrite et les éventuelles modifications nécessaires identifiées par le processus doivent être utilisées.

4.3.4.4 Le plan qualité du système doit contenir des procédures pour la mise à jour du LPD lorsque l'utilisation d'une nouvelle version devient nécessaire.

4.3.4.5 Il convient que les informations sur les erreurs et les défaillances provoquées par le LPD sur d'autres sites et applications, et sur les modifications correspondantes du LPD continuent d'être fournies et analysées formellement pendant la durée de vie.

4.3.3.3.2.4 The rigour of the analysis on feedback of experience shall be consistent with the safety category of the system functions, and the evidence of technical correctness provided by this analysis should be consistent with that achievable when applying IEC 60880.

4.3.3.3.2.5 An evaluation document shall be produced to record

a) that satisfactory feedback of experience compensates for any weakness identified in the evaluation of the design and life cycle of the PDS; or

b) the use of the PDS is rejected because the results of the evaluation are negative, or because the operating experience is not yet sufficient to compensate the weaknesses identified in the development and validation.

4.3.3.4 Comprehensive assessment

4.3.3.4.1 When the evaluations and all the necessary complementary work (modifications of the PDS, complementary tests, complementary documentation) have been completed, a comprehensive justification document for the use of the PDS in the implementation of the system shall be prepared.

4.3.3.4.2 This document, based on the conclusions derived from the evaluations described in 4.3.3.1, 4.3.3.2 and 4.3.3.3, shall record the assessment that demonstrates that the PDS (or part of the PDS) is suitable and has the level of quality appropriate for its intended use in the system and no further modification is needed.

4.3.4 Requirements for integration in the system and maintenance of PDS

4.3.4.1 After the comprehensive assessment, the decision to use the PDS shall be formally made and documented within the project following a formal design review.

4.3.4.2 The procedures for integration of the PDS shall be described in the system quality assurance plan and system integration plan (see 6.2.1 and 6.2.3 of IEC 61513).

4.3.4.3 After acceptance, the PDS shall be placed under the system configuration management (see 6.2.1.2 of IEC 61513) and only the release subjected to the qualification described, and any necessary modifications identified by the process shall be used.

4.3.4.4 The quality plan of the system shall provide procedures for upgrading the PDS when the use of a new release becomes necessary.

4.3.4.5 Information on errors and failures due to the PDS on other sites and applications, and on corresponding modifications of the PDS, should continue to be accessed and formally analysed during the subsequent life.

1 Evaluation de l'aptitude à l'usage (4.3.3.1)

Documentation spécification système	**Documentation d'entrée requise** (4.3.3.1.1)	Spécification du LPD et documentation utilisateur
Comparaison des spécifications du système et du LPD	**Exigences d'évaluation** (4.3.3.1.2)	Identification des modifications et points manquants
Le LPD est apte à l'usage	**Conclusions** Des actions complémentaires sont nécessaires	Le LPD doit être rejeté

2 Evaluation de la qualité (4.3.3.2)

Documentation de conception	**Documentation d'entrée requise** (4.3.3.2.1) Documentation du cycle de vie	(Documentation de l'historique d'exploitation)
Analyse de la conception	**Exigences d'évaluation** (4.3.3.2.2) Analyse de l'AQ	Identification des points manquants
La qualité du cycle de vie du LPD est appropriée ou les modifications nécessaires sont faisables	**Conclusions** Des tests/documentation supplémentaires sont requis ou l'évaluation de l'expérience acquise en exploitation est nécessaire	Le LPD doit être rejeté

3 Evaluation de l'expérience acquise en exploitation (4.3.3.3)

Collecte des données	**Documentation d'entrée requise** (4.3.3.3.1) Temps de fonctionnement	Historique des défauts
	Exigences d'évaluation (4.3.3.3.2)	
L'expérience acquise en exploitation est suffisante	**Conclusions** L'expérience acquise en exploitation n'est pas encore suffisante	Le LPD doit être rejeté

4 Estimation globale (4.3.3.3)

La qualité du LPD est appropriée	Les modifications nécessaires sont faites

5 Intégration dans le système et maintenance (4.3.4)

IEC 2585/2000

Figure 1 – Processus de qualification des logiciels prédéveloppés

1 Suitability evaluation (4.3.3.1)

System specification documentation	**Required input documentation** (4.3.3.1.1)	PDS specification and user's documentation
Comparison of the system and PDS specifications	**Evaluation requirements** (4.3.3.1.2)	Identification of modifications and missing points
The PDS is suitable	**Conclusions** Complementary work is needed	Ought to be rejected

2 Quality evaluation (4.3.3.2)

Design documentation	**Required input documentation** (4.3.3.2.1) Life-cycle documentation	(Operating history documentation)
Analysis of design	**Evaluation requirements** (4.3.3.2.2) Analysis of the QA	Identification of missing points
The quality of the PDS life cycle is appropriate or The needed modifications of the PDS are feasible	**Conclusions** Additional test and documentation is required or Operating experience evaluation required	The PDS ought to be rejected

3 Evaluation of operating experience (4.3.3.3)

Collection of data	**Required input documentation** (4.3.3.3.1) Operating time	History of defects
	Evaluation requirements (4.3.3.3.2)	
Sufficient operating experience	**Conclusions** Operating experience not sufficient yet	The PDS ought to be rejected

4 Comprehensive assessment (4.3.3.3)

The quality of the PDS is appropriate	The needed modifications are done

5 Integration in the system and maintenance (4.3.4)

IEC 2585/2000

Figure 1 – Outline of the qualification process of pre-developed software

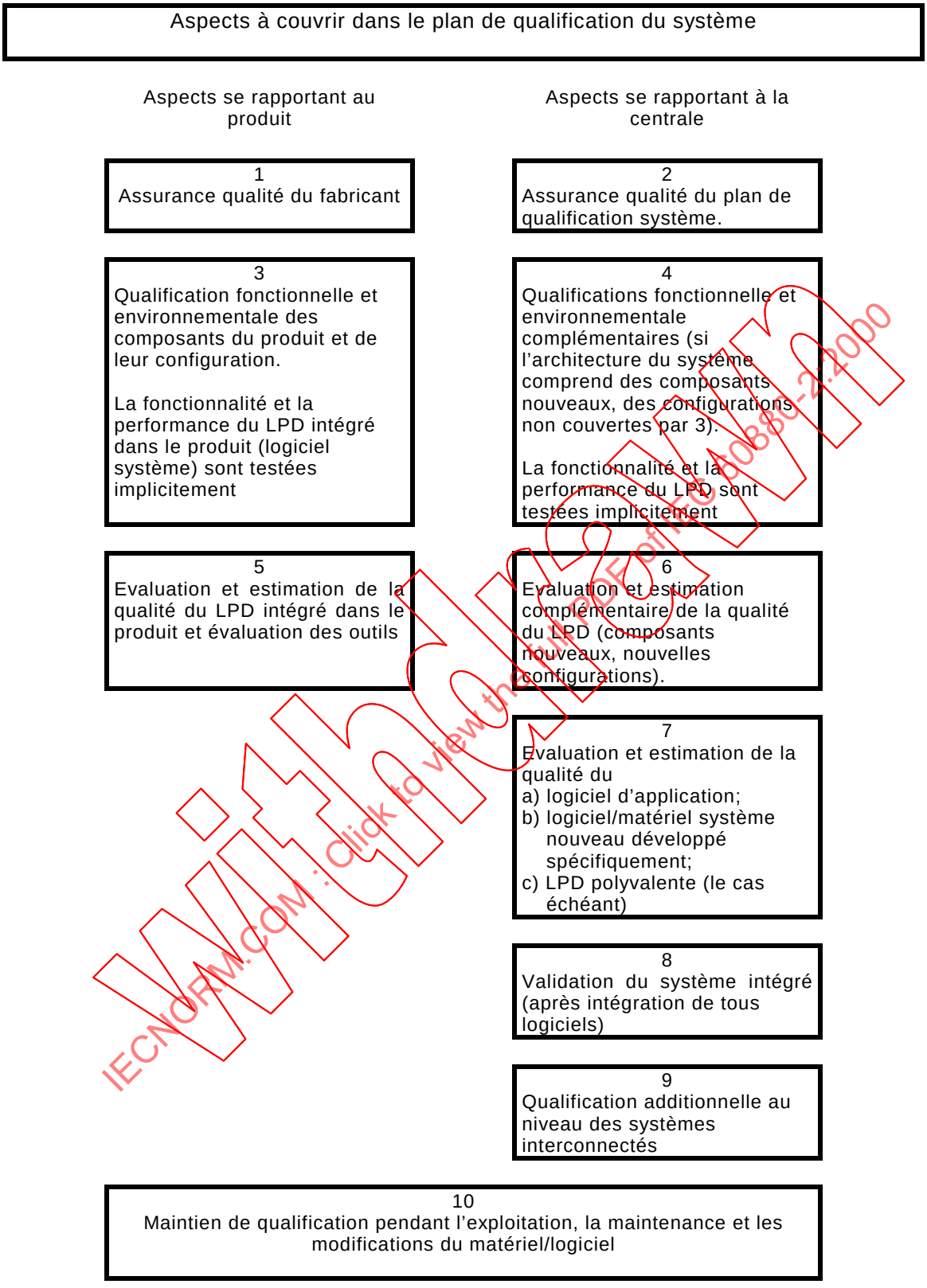
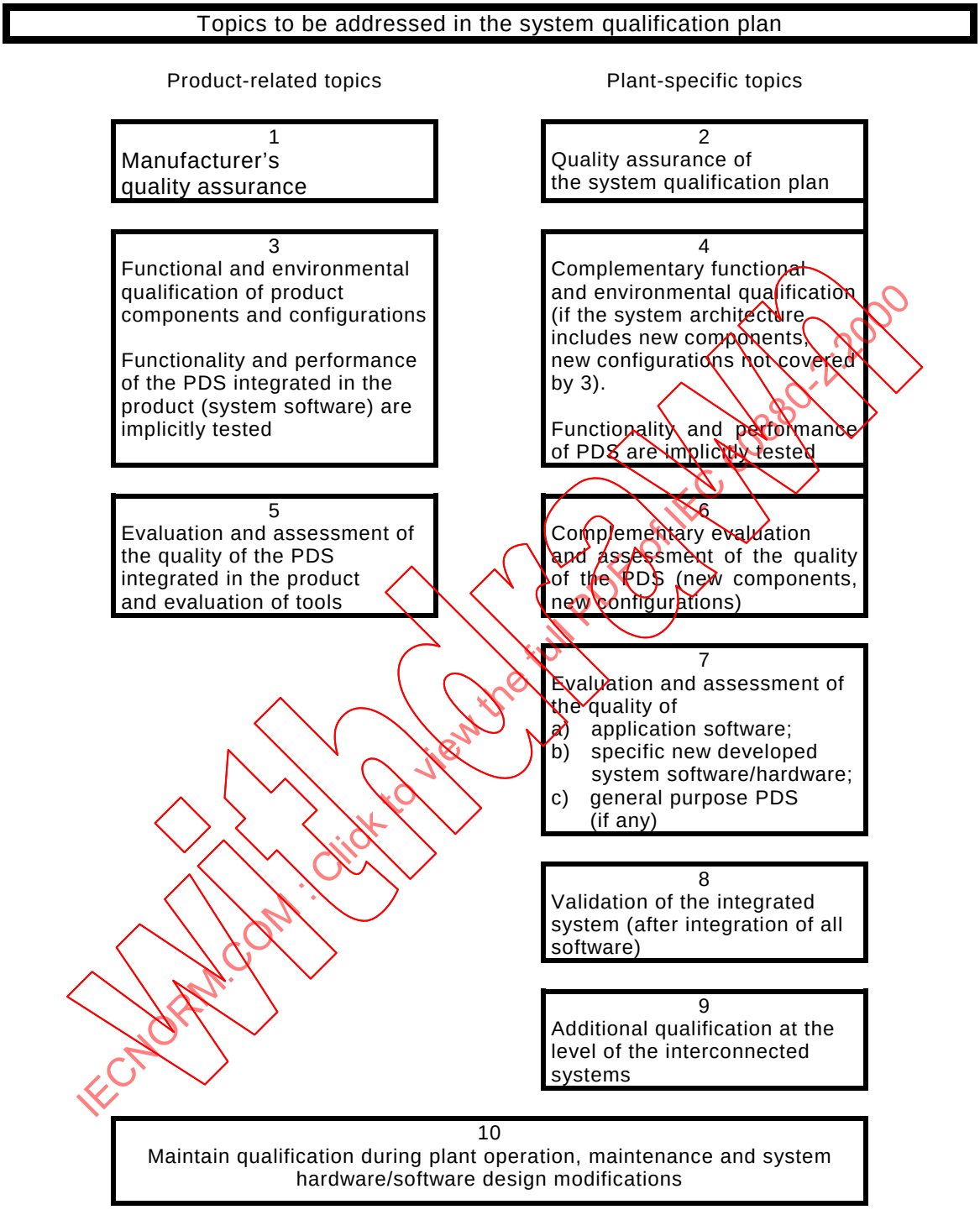


Figure 2 – Relations de l'évaluation et de l'estimation du LPD avec le plan de qualification du système dans lequel il est intégré



IEC 2586/2000

Figure 2 – Relation of PDS evaluation and assessment with the qualification plan of the system in which it is integrated

Annexe A
(informative)

Considérations sur les CCF et la diversification

Les causes et les effets des CCF sont souvent imprévisibles. (Si les CCF étaient prévisibles, il serait possible de les prévenir par une conception appropriée.) Par conséquent, aucune méthode de défense unique n'est suffisante pour se protéger des CCF. Il convient de prendre en considération les méthodes énumérées ci-dessous afin de les combiner et d'offrir une protection relativement globale contre une gamme d'éventuelles causes et d'éventuels effets. Un jugement technique qualifié est requis pour déterminer l'étendue de la protection nécessaire.

A.1 CCF logicielle

Pour qu'il y ait CCF logicielle, des trajectoires de signal ayant pour effet une défaillance identique ou similaire doivent affecter deux systèmes ou voies ou plus (par exemple deux voies de protection, deux régulateurs en boucle fermée ou deux sous-systèmes logiques de commande). Cela s'applique également à deux ou plusieurs systèmes fonctionnellement différents utilisant les mêmes modules logiciels contenant des erreurs.

Pour que les CCF logicielles constituent une préoccupation du point de vue de la sûreté, elles doivent affecter une fonction de sûreté et se produire à un moment où elles pourraient causer un risque pour la sûreté, ou doivent elles-mêmes entraîner un risque pour la sûreté comme par exemple une perte de protection ou de commande. Cette période de temps peut être longue dans le cas de défauts non révélés, ou si le même logiciel est utilisé sur deux systèmes dissemblables (par exemple la mise en oeuvre de deux fonctions de régulation différentes). La période concernée peut être longue si une défaillance apparaît suite à un état matériel inhabituel ou à une défaillance matérielle inhabituelle, susceptible d'apparaître pour des raisons administratives ou opérationnelles engendrant une trajectoire de signal imprévue.

Si des logiciels, des méthodes de mise en oeuvre ou des algorithmes identiques ou similaires sont utilisés dans des systèmes redondants ou différents, il existe un élément commun significatif.

Il n'existe actuellement aucune méthode convenue d'estimation de la probabilité de défaillance ou du taux de défaillance des logiciels.

A.2 Causes et effets des CCF potentielles

a) Potentiel de CCF

Il peut exister un potentiel de CCF logicielles entre différents systèmes ou entre des voies différentes d'un système lorsque des modules logiciels communs sont utilisés. Les autres caractéristiques communes présentant un potentiel de CCF sont l'architecture, les algorithmes, les méthodes de développement, les outils, les méthodes de mise en oeuvre, le personnel et la gestion.

Les prescriptions mal comprises ou transformées de manière incorrecte peuvent entraîner des défauts des spécifications du logiciel débouchant sur des risques de CCF logicielles. Les déficiences du logiciel peuvent être dues à des prescriptions logicielles et à des spécifications logicielles incorrectes, incomplètes, imprécises ou mal comprises. Des erreurs de conception ou des défauts logiciels peuvent être présents dans les programmes diversifiés en raison de facteurs humains communs tels que la formation, l'organisation, les processus de pensée et les approches de conception.

Annex A
(informative)

Considerations of CCF and diversity

CCF causes and effects are often unpredictable. (If CCF causes could be predicted, it would be possible to prevent them by design measures.) Consequently, no single defence measure is sufficient to protect against CCF. The defence measures listed should be considered for use in combination to provide relatively broad coverage against a range of possible causes and effects. The exact breadth needed is a matter of engineering judgement.

A.1 CCF due to software

For a software induced CCF, signal trajectories must exercise a software fault causing a failure that affects two or more systems or channels (for example, two protection channels, two closed loop controllers, or two logic control subsystems). This applies also to two or more functionally different systems that use the same software modules containing faults.

For the CCF to be of safety concern, these failures must disable a safety function and happen within a time period in which a safety challenge could result, or it must itself cause a safety challenge such as loss of protection or control. This time period may be long if unrevealed faults are considered, or if the same software is used in two dissimilar systems performing the same function (for example, the implementation of two different control functions). The time period may also be long if a failure results from an unusual hardware condition or appears for administrative or operational reasons, causing an unexpected signal trajectory.

If the same or similar software, implementation methods or algorithms are used in redundant or in different systems, then a significant common element exists.

No agreed method of estimation currently exists for estimating the probability of failure or for failure rate arising from software faults.

A.2 Potential CCF causes and effects

a) CCF potential

A potential for a software-induced CCF of different systems or between different channels in one system exists if common software or software modules are used. Other common features with CCF potential include common architecture, algorithms, development methods, tools, implementation methods, staffing and management.

Requirements that are not properly understood or not correctly transformed can result in faults in the software specification resulting in risks of CCF due to exercising the resulting software fault. Deficiencies in software can be due to incorrect, incomplete, inaccurate or misunderstood software requirements and software specifications. Design errors leading to software faults can be introduced into diverse programs, due to common human factors such as training, organization, thinking processes and design approaches.

Une autre cause potentielle de CCF peut être la connexion de systèmes avec d'autres systèmes contenant des logiciels de moindre qualité.

b) Trajectoires de signal

Les trajectoires de signal (voir 3.19) peuvent entraîner des CCF lorsqu'elles sont lues

- par chacune des voies redondantes d'un système utilisant du logiciel commun ou similaire;
- par deux systèmes dont les fonctions sont diversifiées et qui ont du logiciel commun.

Un défaut logiciel peut entraîner une défaillance logicielle lorsqu'une trajectoire de signal spécifique apparaît. Si cette trajectoire de signal est similaire pour deux ou plusieurs voies ou systèmes, cela peut entraîner une CCF. Les défaillances de cause commune compromettront une ou plusieurs couches de défense lorsqu'une qualité, une indépendance et une diversification suffisantes n'auront pas été assurées.

c) Anomalies et événements anormaux

Des défaillances matérielles, des anomalies de la centrale et des événements anormaux peuvent entraîner des trajectoires de signal imprévisibles, ainsi que des états logiciels, des transitoires ou des états de surcharge imprévus, non couverts par les prescriptions initiales ou par la conception du logiciel.

Les événements potentiels qui entraînent des CCF sont:

- les défaillances de signal de temporisation commun, entraînant la perte des actions temporisées;
- les transitoires d'alimentation entraînant l'arrêt ou le redémarrage automatique du logiciel;
- les arrêts d'urgence de la centrale causant la surcharge des voies de communication;
- la saturation des capacités de l'opérateur, le conduisant à réagir incorrectement;
- les demandes de l'opérateur saturant la capacité du système en cas d'arrêt d'urgence de la centrale;
- toutes les fonctions des régulateurs engagées et en fonctionnement, et
- les situations anormales pendant les arrêts et la mise en service.

A.3 Défense contre les CCF

Les moyens de défense possibles sont:

- les méthodes utilisées pendant toute la durée de vie dans le but de produire un logiciel exempt de défauts (voir 4.1.2);
- la démonstration et l'amélioration de la qualité du logiciel commun;
- l'utilisation du logiciel commun dans des conditions d'exploitation très restreintes et garanties;
- les limitations des effets des défaillances logicielles;
- la conception des canaux ou systèmes telle qu'une défaillance simultanée de deux canaux ou systèmes est très improbable car il est démontré que les trajectoires des signaux des systèmes sont différentes;
- la conception des canaux ou systèmes utilisant un fonctionnement asynchrone. Ceci peut être utilisé pour montrer une défense vers les même processeurs dans des canaux différents soumis à des trajectoires identiques au même temps; et
- la diversification pour certaines ou l'ensemble des fonctions, et l'amélioration des concepts d'indépendance pendant toute la durée de vie (voir 4.1.4).

Another potential cause of CCF could result from connection of systems to ones with lower quality software.

b) Signal trajectories

The signal trajectories (see 3.19) can cause a CCF when they are read

- by each redundant channel of a system using common software;
- by two systems whose functions are diverse but which use common software.

A software fault may result in a software failure when a specific signal trajectory appears. If this signal trajectory is identical for two or more channels or systems, this may result in a CCF which will jeopardize one or more defence layers when sufficient quality, independence and diversity are not provided.

c) Abnormal conditions and events

Abnormal hardware failures, plant conditions and events can cause unforeseen signal trajectories, unexpected software states, transients or overload conditions that were not covered by the initial requirements or by the software design.

Potential events which may cause CCF include:

- common timing signal failure, causing loss of timed actions;
- power supply transients causing software stop or auto-restart;
- plant trips causing communication channels to overload;
- saturation of operator capacity, causing an incorrect action;
- operator demands saturating system capacity during plant trips and transients;
- all automatic controller functions engaged and operating; and
- abnormal conditions during outages and commissioning.

A.3 CCF defences

Possible defence features include:

- methods used throughout the life cycle which aim to produce fault-free software (see 4.1.2);
- demonstration and enhancement of the quality of common software;
- use of common software in a very narrow and guaranteed set of operating conditions;
- limitations of the effects of software failures;
- design of the channels or systems such that a coincident failure of two channels or systems is very unlikely due to there being a demonstrable difference in the signal trajectories for the systems;
- design of the channels or systems using asynchronous operation; this may be used to show defence against the same processors in different channels being subjected to identical trajectories at the same time; and
- diverse features for some or all functions and enhancement of independence concepts during the whole life cycle (see 4.1.4).

A.4 Preuve de conformité

Les méthodes permettant la preuve de la conformité sont les suivantes:

- l'utilisation de méthodes formelles;
- la réutilisation de modules logiciels standards validés avec des interfaces claires et validées (voir 4.3); les fonctions typiques comprennent par exemple les modules pour le pilotage des périphériques (voir la CEI 60880), la surveillance des processus et l'acquisition des signaux d'entrée, les algorithmes de base de régulation (comme par exemple proportionnelle-intégrale-dérivée, bande morte, hystérésis);
- l'utilisation d'outils et de procédures indépendants des processus de décodage du code chargé en mémoire et la démonstration que le code chargé est conforme à la spécification;
- l'utilisation de l'analyse statique du code afin d'identifier les flux de commandes et de données, et de démontrer que les processus de prise de décision et les processus logiques sont corrects;
- l'utilisation de deux versions logicielles testées dos-à-dos, en soumettant le logiciel à des trajectoires de signal aléatoires. Cette méthode peut être utilisée en complément aux tests systématiques pour la détection des défauts de conception et codage;
- la réalisation d'un programme de tests poussé et progressif, où le fonctionnement correct de chaque composant du système est rigoureusement validé avant d'être intégré dans le système.

A.5 Caractéristiques de la diversité

a) Les caractéristiques importantes de la diversité logicielle sont:
   - la diversité fonctionnelle;
   - des spécifications de conception différentes pour le même besoin fonctionnel;
   - la mise en oeuvre différente de fonctions (N versions de logiciel) pour la même spécification.

b) La diversité au niveau système peut comprendre:
   - l'utilisation de systèmes indépendants pour différents critères d'action;
   - l'utilisation de technologies de base différentes, comme par exemple des calculateurs par rapport à des conceptions câblées;
   - l'utilisation de différents types de calculateurs, modules matériels et concepts principaux de conception;
   - l'utilisation de différentes classes de techniques informatiques telles que les automates, les microprocesseurs ou les mini-ordinateurs.

c) Les caractéristiques de l'approche de conception et les solutions aux problèmes qui améliorent la diversité considèrent des différences en matière de
   - algorithmes de traitement;
   - données de configuration, d'étalonnage et fonctionnelles;
   - matériel d'acquisition de signaux;
   - interfaces matérielles et communications;
   - processus d'échantillonnage des entrées;
   - chronologie d'opérations;
   - processus de temporisation;
   - l'utilisation d'informations mémorisées, de verrouillages et de taux de variation.

A.4 Demonstration of correctness

Methods of supporting demonstration of correctness include:

- the use of formal methods;
- reuse of proven software standard modules with a clear and proven interface (see 4.3); typical functions include, for example, modules for device driving (see IEC 60880), process monitoring and input gathering, basic control algorithms (such as proportional-integral-derivative, deadband, hysteresis);
- use of tools and procedures independent of the design processes for decoding of the code loaded in memory and demonstration that the loaded code matches the specification;
- the use of static analysis of code to identify control and data flow, and to demonstrate correct decision and logic processes;
- the use of two software versions tested back to back, exercising the software by random signal trajectories. This method can be used in addition to systematic testing for detection of design and coding faults;
- performing a comprehensive bottom-up testing programme, where the correct operation of each system component is thoroughly validated before being integrated into the system.

A.5 Diversity features

a) Software diversity features of importance include:
   - functional diversity;
   - different design specifications for the same functional requirements;
   - implementing the functions differently (N-version software) for the same specification.

b) Diversity at the system level can include:
   - use of independent systems for different actuation criteria;
   - use of different basic technology, such as computers versus hardwired design;
   - use of different types of computers, hardware modules and major design concepts;
   - use of different classes of computer technique such as PLCs, microprocessors or mini-computers.

c) The design approach features and problem solutions which enhance diversity include differences of
   - processing algorithms;
   - data for configuration, calibration and functionality;
   - signal input hardware;
   - hardware interfaces and communications;
   - input sampling processes;
   - time sequences of operations;
   - timing processes;
   - use of historical information, latches and rates of change.

d) Les différences de méthodes de conception et de mise en oeuvre comprennent:

- les langages;
- les systèmes de compilation;
- les bibliothèques supports;
- les outils logiciels;
- les techniques de programmation;
- les logiciels systèmes et applicatifs;
- les structures logicielles;
- l'utilisation différente des mêmes modules logiciels;
- les données et les structures de données.

e) Diversité pendant les tests (tests dos-à-dos)

f) Les divers aspects de l'approche de la gestion incluent:

- deux conceptions selon des méthodes de développement volontairement dissemblables (forcées);
- la séparation des équipes de conception;
- la restriction de la communication entres les équipes;
- la communication formelle de la levée des ambiguïtés dans les prescriptions ou les spécifications;
- l'utilisation de processus de définition logique différents;
- les différences dans les méthodes de documentation;
- l'utilisation de personnel différent.

A.6 Inconvénients, avantages et justification de la diversité

a) Inconvénients

Les inconvénients introduits par la diversité peuvent être les suivants:

- complexité globale plus grande;
- risque de mise en marche intempestive accru;
- spécifications et conception plus complexes;
- contrôle de deux fournisseurs;
- problèmes de maintenance et de modification, par exemple pour assurer que la diversité n'est pas perdue lors d'une modification;
- documentation plus importante;
- besoins en espace, en fournitures et de contrôle de l'environnement plus importants;
- le coût de plusieurs versions du logiciel peut réduire son potentiel commercial, sauf en tant que méthode de test;
- la complexité de la technique des blocs de recouvrement peut réduire la fiabilité;
- chaque version produite peut être de qualité inférieure.

b) Avantages

Lorsque la diversité fonctionnelle ou logicielle est utilisée, des versions à diversité adéquate assurent une plus grande protection contre les CCF provoquées par le logiciel.

c) Justification

La justification peut porter sur l'amélioration de la fiabilité des fonctions de sûreté obtenues par l'utilisation de la diversité.

d) Differences in design and implementation methods include:

- languages;
- compilation systems;
- support libraries;
- software tools;
- programming techniques;
- system and application software;
- software structures;
- different use of the same software modules;
- data and data structures.

e) Diversity during tests (back-to-back testing)

f) Diverse aspects of management approach include:

- two designs following deliberately dissimilar development methods (forced);
- separation of the design teams;
- restriction of communication between the teams;
- formal communication of resolution of ambiguities in requirements or specifications;
- use of different logic definition processes;
- differences in documentation methods;
- use of different staff.

A.6 Drawbacks, benefits and justification of diversity

a) Drawbacks

The disadvantages introduced by diversity may include:

- greater overall complexity;
- increased risk of spurious actuation;
- more complex specifications and design;
- control of two suppliers;
- maintenance and modification problems, for example, ensuring diversity is not lost during modification;
- increased documentation;
- increased space, supplies, environmental control requirements;
- the cost of several versions of the software can reduce its commercial potential, except as a testing method;
- recovery block technique complexity can reduce reliability;
- each version produced may be of lower quality.

b) Benefits

When functional or software diversity are used, adequately diverse versions give increased protection against CCF due to software.

c) Justification

The justification can consider the improvement in reliability of the safety functions achieved by use of diversity.

Annexe B
(informative)[1)]

Prescriptions de la CEI 60880 pour l'utilisation et la qualification des outils logiciels

Les prescriptions pour les outils selon la CEI 60880 sont les suivantes.

1) L'utilisation d'un langage formel de spécification peut être une aide pour démontrer la cohérence et l'exhaustivité des exigences fonctionnelles pour le logiciel. Des outils automatiques peuvent être utilisés à cet effet (voir CEI 60880, 4.10).

2) Dans le cas d'utilisation d'un logiciel commercialisé, il convient que la démonstration de son bon fonctionnement soit réalisée (voir CEI 60880, 5.1.2e)).

3) Il convient d'utiliser des langages disposant d'un traducteur complètement testé. Si un traducteur partiellement testé est utilisé, une vérification supplémentaire devra montrer que le résultat de la traduction est correct (voir CEI 60880, 5.2.1).

4) De même que les points spécifiques mentionnés dans l'annexe D de la CEI 60880, il est recommandé qu'un langage de programmation d'un système de sûreté et son traducteur n'interdisent pas par leur conception
   - les constructions limitant les erreurs;
   - les vérifications des types lors de la traduction;
   - la vérification des types et des limites de validité des pointeurs de tables, ainsi que la vérification des paramètres lors de l'exécution (voir CEI 60880, 5.2.4).

5) Des aides automatiques de test doivent être disponibles (voir CEI 60880, 5.2.5).

6) Il est recommandé d'utiliser des outils automatiques (voir CEI 60880, 5.2.6).

7) Des mesures d'assurance qualité doivent être établies pour les outils logiciels utilisés dans la vérification du système intégré, en rapport avec l'importance de ces outils pour cette action (voir CEI 60880, 7.5).

8) Les outils matériels et logiciels utilisés pour la validation du système informatique ne nécessitent pas de vérification spéciale. Il convient cependant de pouvoir montrer qu'ils remplissent leur rôle (voir CEI 60880, article 8).

9) Autant que possible, des outils de développement automatisés devraient être utilisés (voir CEI 60880, B1.bi).

10) Il convient que le traducteur (et l'interpréteur, le compilateur croisé, l'émulateur), l'éditeur de liens et le chargeur soient largement testés avant leur utilisation; leur fonctionnement est considéré comme très important (voir CEI 60880, D1.a).

11) En ce qui concerne le traducteur, l'éditeur de liens et le chargeur, il est recommandé que des données de fiabilité de qualité suffisante soient disponibles (voir CEI 60880, D1.b).

12) Dans le cas où des programmes auxiliaires du système sont utilisés tels que des aides, des systèmes de documentation ou équivalent, il convient qu'ils soient testés de manière appropriée, avant l'utilisation (voir CEI 60880, D1.c).

13) Il est recommandé d'utiliser, dans la mesure du possible, des outils automatiques de tests dans la génération des différents cas de test (voir CEI 60880, article E2).

---

1) Pour les prescriptions normatives, voir la CEI 60880.

Annex B
(informative)[1)]

IEC 60880 requirements for the use and qualification of software tools

The requirements on tools taken from IEC 60880 are the following.

1) The use of a formal specification language may be a help to show coherence and completeness of the software functional requirements. Automatic tools may be used for this purpose (see IEC 60880, 4.10).
2) Where standard software from a manufacturer or supplier is used, it should be shown to have operated satisfactorily (see IEC 60880, 5.1.2e)).
3) Languages with a thoroughly tested translator should be used. If no thoroughly tested translator is employed, additional verification shall show that the result of the translation is correct (see IEC 60880, 5.2.1).
4) As well as the specific points mentioned in appendix D of IEC 60880, a programming language for safety systems and its translator should not prevent by their design
   - error-limiting constructs;
   - translation-time type checking;
   - run-time type and array bound check, and parameter checking (see IEC 60880, 5.2.4).
5) Automatic testing aids shall be available (see IEC 60880, 5.2.5).
6) The use of automatic tools is recommended (see IEC 60880, 5.2.6).
7) Quality assurance measures shall be established for software tools used for integrated system verification, commensurate with the importance of those tools for verification (see IEC 60880, 7.5).
8) Hardware and software tools used for computer system validation need no special verification. They should, however, be shown to be suited to their purpose (see IEC 60880, clause 8).
9) As far as possible suitably qualified automatic development aids should be used (see IEC 60880, B1.bi).
10) Translator (also interpreter, cross-compiler, emulator), linkage editor and loader should be thoroughly tested prior to use; operation is considered very important (see IEC 60880, D1.a).
11) Reliability data of sufficient quality about translator, linkage editor and loader should be available (see IEC 60880, D1.b).
12) In cases where auxiliary system programs are used, such as aids, documentation systems and the like, they should be appropriately tested before being employed (see IEC 60880, D1.c).
13) Automatic testing tools should be used as much as possible in deriving test cases (see IEC 60880, clause E2).

---

1) For normative statements, see IEC 60880.

Annexe C
(informative)

Outils pour la production et la vérification des spécifications, de la conception et du code

Les outils constituent une partie essentielle du développement des logiciels qui réalisent des fonctions de sûreté. Des méthodes qui sont appliquées strictement manuellement sont fortement sujettes aux erreurs et exigent l'intervention de personnes très bien formées. De ce fait, il convient qu'elles soient aidées par des outils utilisant les techniques mathématiques afin de révéler la structure et les relations fonctionnelles internes du logiciel et de vérifier la cohérence interne, la cohérence avec un éventuel modèle antérieur, les propriétés désirables/ indésirables, etc.

La démonstration finale de conformité du code à la spécification peut être effectuée au moyen d'un analyseur de conformité. Lorsqu'un générateur de code éprouvé assure que le code exécutable est entièrement cohérent avec la description de conception, les analyses statique et dynamique du code permettent un contrôle diversifié de la conformité de cette description.

Les outils destinés aux méthodes formelles de spécification et de conception se décomposent en outils constructifs et en outils analytiques.

C.1 Outils constructifs

Les outils constructifs sont utilisés pour produire la spécification, la conception et le code, et peuvent comprendre:

a) Editeur de texte

Du fait que les méthodes formelles basées sur la théorie des ensembles, le calcul des prédicats et des propositions, exigent des symboles mathématiques spéciaux, il est important qu'un éditeur de texte approprié soit disponible, capable de les afficher sur un écran à haute définition et de les imprimer de manière lisible.

b) Interface graphique

Une capacité graphique appropriée est requise lorsque la méthode formelle implique l'utilisation de graphiques.

c) Générateur automatique de code

Une fois qu'une spécification formelle a été validée, l'intégrité du processus de conception peut être grandement améliorée par l'utilisation d'un générateur automatique de code validé. Un tel générateur de code permet de transformer la spécification en code exécutable et donc de réduire les possibilités d'introduction d'erreurs. En outre, un sous-ensemble fiable d'un langage peut être mis en oeuvre par conception du générateur de code.

Il convient que des modules logiciels certifiés soient utilisés pour les fonctions standards.

Il convient que le code généré de manière automatique soit lisible. Il convient que les commentaires permettent l'identification des parties associées de la spécification. Il convient que la structure du code généré de manière automatique permette la vérification automatique.

d) Générateur d'obligations de preuve

Les méthodes formelles basées sur le raisonnement logique exigent un générateur d'obligations de preuve qui enregistre automatiquement les obligations de preuve inhérentes aux étapes de conception.

Annex C
(informative)

Tools for production and checking of specification, design and code

Tools now form an essential part of the development environment for software performing safety functions. Methods which are applied purely manually are highly error-prone and require the involvement of very well trained humans. Therefore, they should be supported by tools which use mathematical techniques to reveal the structure and internal functional relationships of the software and to check for internal consistency, consistency with some prior model, desirable/undesirable properties, etc.

Final demonstration that the code meets its specification can be performed by means of a compliance analyser. Where a proven code generator ensures that the executable code is fully consistent with its design description, then static and dynamic analyses of the code provide a diverse check on the correctness of that description.

Tools for formal specification and design methods can be classified as constructive or analytical tools.

C.1 Constructive tools

Constructive tools are used to support the specification, design and code phases of development, and may include:

a) Text editor

Because formal methods based on set theory and on predicate and propositional calculus require special mathematical symbols, it is important that a suitable text editor is available capable of both displaying these on a high definition screen and also printing them out legibly.

b) Graphical interface

A suitable graphics capability is required where the formal method involves the use of graphics.

c) Automatic code generator

Once a formal specification has been proven, the integrity of the design process can be greatly enhanced by the use of a validated automatic code generator. Such a code generator will transform the specification into executable code and thus reduce the likelihood of introducing errors. Additionally, a safe subset of a language may be enforced through the code generator design.

Certified software modules should be used for standard functions.

Automatically generated code should be readable. Comments should support the identification of the associated parts of the specification. The structure of automatically generated code should support automatic verification.

d) Proof obligation generator

Formal methods based on logical reasoning require a proof obligation generator that automatically registers the proof obligations arising during the design steps.

C.2 Outils analytiques

Les outils analytiques permettent le contrôle de la spécification, de la conception et du code. Ils peuvent comprendre:

a) Contrôleur syntaxique

Un contrôleur de syntaxe donne des informations sur la structure du programme, l'utilisation des données du programme, la dépendance des variables de sortie vis-à-vis des variables d'entrée, et le flux de contrôle dans le programme, ce qui permet

1) d'identifier les défauts de structure comme les démarrages multiples, les fins multiples, le code inaccessible, le code redondant, la non-utilisation des résultats de fonctions;
2) d'identifier la hiérarchie des modules/sous-programmes;
3) d'identifier la violation des normes et des conventions de programmation, y compris les vérifications de branchements inconditionnels dans des boucles;
4) d'identifier les données qui sont lues avant d'être écrites, les données écrites avant d'être lues, les données écrites deux fois sans lecture intermédiaire;
5) de contrôler le flux des informations par rapport à la spécification;
6) d'aider à la conception d'un plan de test dynamique;
7) d'effectuer la gestion et la génération éventuelle des données de test.

b) Contrôleur sémantique

Un contrôleur sémantique décrit les relations mathématiques entre les variables de sortie et d'entrée pour chacun des chemins possibles du point de vue sémantique dans les régions sans branchements du programme. Cela permet de vérifier ce que fera le programme dans toutes les circonstances et de détecter des défauts tels que les valeurs de sorties inattendues affectées par les valeurs d'entrée, la réponse incorrecte à des valeurs inattendues d'entrée, le signe incorrect de fonctions et d'opérateurs, etc.

c) Générateur de preuves formelles

Les preuves formelles d'une conception exigent l'utilisation d'un programme interactif qui effectue les manipulations de symboles nécessaires sous le contrôle d'un opérateur humain afin de remplir les obligations de preuve. Un tel programme s'appelle un assistant démonstrateur de théorèmes (TPA). Les TPA sont de gros programmes dont l'absence d'erreur ne peut pas être démontrée, et qui exigent donc l'emploi d'un moyen de preuve diversifié qu'il convient d'utiliser. Cela implique en général l'application d'un contrôleur de preuve dont l'entrée est la sortie du TPA. Il convient que le générateur de preuves soit basé sur une théorie de preuve formelle et il convient qu'il soit vérifié par rapport à cette théorie de preuve.

d) Animateur

Il convient que chaque fois que possible les spécifications formelles soient animées de manière que l'utilisateur final du système puisse examiner des aspects de la spécification ou de la conception afin de valider (dans toute la mesure du possible) la conception proposée. Il convient que l'animation soit aussi représentative que possible de la conception et il se peut qu'elle exige l'utilisation de prototypes pour démontrer les aspects non fonctionnels. Cette évaluation est faite par rapport aux critères de l'utilisateur et les prescriptions du système peuvent être modifiés à la lumière de cette évaluation.

e) Analyseur de conformité

Un analyseur de conformité peut démontrer que le code met correctement en oeuvre la spécification. Pour cette démonstration, cet outil utilise des pré-conditions et des post-conditions plus des invariants de boucle. L'outil confirme systématiquement que chaque condition est remplie par le code.