



**PROJET DE GUIDE
D'APPLICATION
DE LA
RÉGLEMENTATION**

C-138 (F)

**LOGICIELS DE SYSTÈMES DE
PROTECTION ET DE CONTRÔLE**

Publié pour commentaires par la
Commission de contrôle de l'énergie atomique
Octobre 1999



Commission de contrôle
de l'énergie atomique

Atomic Energy
Control Board

Canada

PROJET DE GUIDE D'APPLICATION DE LA RÉGLEMENTATION

**Logiciels de systèmes de protection et de contrôle
C-138 (F)**

**Publié pour commentaires par la
Commission de contrôle de l'énergie atomique
October 1999**

Documents d'application de la réglementation de la CCEA

La Commission de contrôle de l'énergie atomique (CCEA) fonctionne à l'intérieur d'un cadre juridique constitué de la législation et, à l'appui, de documents d'application de la réglementation. Le terme « législation » renvoie à différents instruments légaux exécutoires : des lois, des règlements, des permis et des directives. Quant aux documents d'application de la réglementation – des politiques, des normes, des guides, des avis, des procédures et des documents d'information –, ils soutiennent et expliquent davantage ces instruments. Les activités de réglementation de la CCEA reposent sur ces instruments et ces documents.

Voici les principales classes de documents de réglementation de la CCEA :

- **Politique d'application de la réglementation** : document qui décrit la doctrine, les principes et les facteurs fondamentaux utilisés par la CCEA dans son programme d'application de la réglementation
- **Norme d'application de la réglementation** : document qui peut servir à une évaluation de conformité et qui décrit les règles, les caractéristiques ou les pratiques que la CCEA accepte comme conformes aux exigences réglementaires.
- **Guide d'application de la réglementation** : document qui sert de guide ou qui décrit des caractéristiques ou des pratiques recommandées par la CCEA et qui, d'après elle, permettent de respecter les exigences réglementaires ou d'améliorer l'efficacité administrative.
- **Avis d'application de la réglementation** : document qui contient des conseils et des renseignements propres à un cas donné et qui sert à alerter les titulaires de permis et d'autres personnes à propos d'importantes questions de santé, de sûreté ou de conformité auxquelles il faut donner suite en temps utile.
- **Procédure d'application de la réglementation** : document qui décrit les modalités de travail qu'utilise la CCEA pour administrer les exigences réglementaires dont elle est responsable.

Les politiques, normes, guides, avis et procédures d'application de la réglementation ne créent pas d'exigences exécutoires, mais étayent les exigences réglementaires des règlements, des permis et des autres instruments exécutoires. Cela dit, quand il y a lieu, un document d'application de la réglementation peut être transformé en instrument exécutoire par incorporation dans un règlement de la CCEA, un de ses permis ou un autre instrument exécutoire créé en vertu de la *Loi sur le contrôle de l'énergie atomique*.

PROJET DE GUIDE D'APPLICATION DE LA RÉGLEMENTATION

Logiciels de systèmes de protection et de contrôle

C-138 (F)

October 1999

AVIS

Le 20 mars 1997, le projet de loi C-23, *Loi sur la sûreté et la réglementation nucléaires*, recevait la sanction royale. Cette loi n'est pas encore entrée en vigueur. Lorsque la nouvelle Loi et ses règlements d'application auront été promulgués, le présent document sera révisé. Le Guide d'application de la réglementation C-138 (F) a été préparé sous le régime de la loi actuelle, *Loi sur le contrôle de l'énergie atomique*, et des règlements d'application, lesquels textes demeurent en vigueur à une date fixée en 2000 par décret du gouverneur en conseil.

A propos du document

Commentaires

Afin de nous permettre de déterminer l'impact et la valeur de ce document au moyen d'expérience pratique, nous vous invitons à envoyer vos commentaires au cours de la période d'essai prolongée. À la fin de cette période de deux mois, nous les étudierons pour déterminer la meilleure façon d'améliorer le document. Nous vous prions donc de nous faire parvenir vos commentaires par écrit avant le 30 décembre 1999, en précisant le numéro de dossier 1-8-8-138.

Consultation du document

Vous pouvez consulter le document C-138 (en anglais et en français) à partir du site web de la CCEA : www.aecb-ccea.gc.ca ou le commander en communiquant avec la personne suivante :

Adjointe aux opérations
Section de la documentation de la Commission
Commission de contrôle de l'énergie atomique
C. P. 1046, Succursale B, 280, rue Slater
Ottawa ON K1P 5S9
CANADA

Téléphone (613) 996-9505
Télécopieur (613) 995-5086
Courriel : reg@atomcon.gc.ca

Table des matières

A propos du document	i
Commentaires	i
Consultation du document	i
But	1
Portée	1
Classification	2
1. Exigences techniques	2
1.1 Exigences du logiciel	2
1.2 Inspection systématique de la conception et de la mise en oeuvre du logiciel	4
1.3 Essai du logiciel	6
2. Processus de développement des logiciels, gestion et assurance de la qualité	8
2.1 Plans, normes et procédures	8
2.2 Gestion de la configuration	8
2.3 Processus de développement des logiciels	9
2.4 Emploi de logiciels déjà développés	12
3. Considérations relatives au système	13
3.1 Spécification du système	14
3.2 Conception du système	14
3.3 Interface matériel-logiciel	15
Glossaire	18
Références	22

But

Le présent guide d'application de la réglementation établit ce qui constitue :

- une description appropriée des logiciels utilisés dans les systèmes de protection, de commande et de contrôle
- une démonstration appropriée du caractère adéquat de la conception d'un logiciel.

Le guide n'est pas une norme et il n'est pas censé servir de norme pour l'élaboration de logiciels ou l'assurance de leur qualité. Il vise à aider les employés de la CCEA à évaluer des logiciels, et les demandeurs ou titulaires de permis à préparer leurs documents à être présentés à la CCEA.

Portée

Le guide vise spécifiquement le volet logiciel des systèmes de protection, de commande et de contrôle, ce qui comprend les logiciels installés à titre de microprogrammes. L'ensemble du système et les autres composantes sont également importants, mais le guide ne contient pas d'instructions détaillées quant à leur description ou à la démonstration de leur valeur, car ces aspects débordent de son propos. Cependant, la section 4 traite du rapport entre le logiciel et le système dans son ensemble, des interfaces entre le logiciel et les autres composantes ainsi que de l'interface avec l'ordinateur sur lequel tourne le logiciel.

Le guide ne vise pas les logiciels non exploités en temps réel et utilisés pour la conception et la mise au point d'autres systèmes.

Le guide vise les logiciels exploités dans des systèmes qui :

- fournissent une protection contre le rejet indésirable de matières radioactives ou de radioactivité
- contrôlent les réactions nucléaires, le mouvement de matières radioactives ou le refroidissement de matières radioactives
- atténuent les effets de défaillances d'autres systèmes, de manière que soit réduit le risque d'exposition au rayonnement.

Le guide vise les logiciels nouveaux, y compris :

- les logiciels installés dans de nouveaux systèmes
- les logiciels qui donnent suite à de nouvelles exigences dans des systèmes existants
- les logiciels qui remplacent des logiciels existants.

Les changements apportés à des logiciels existants et destinés à corriger ou à adapter une fonctionnalité existante doivent faire l'objet d'un contrôle rigoureux. De tels changements sont

l'occasion d'améliorer la documentation logicielle existante et de la faire correspondre plus intimement aux critères de qualité établis dans les normes et dans le guide.

Le guide ne cite pas de règlements ni de conditions de permis en vertu desquels des systèmes ou des modifications à des systèmes doivent être approuvés par la CCEA avant l'installation. Il se limite à établir comment un logiciel doit être décrit et comment on doit faire la démonstration de son caractère adéquat, si une telle description ou démonstration est requise en vertu de la réglementation actuelle et de conditions de permis.

Classification

Les demandeurs ou les titulaires de permis peuvent classer les logiciels dans différentes catégories en fonction du niveau de sûreté du système et du rôle que jouent les logiciels en égard à la sûreté du système. La CCEA ne prescrit pas de méthode de classification des systèmes utilisant des logiciels, mais on suppose dans le présent guide qu'il existe de telles catégories. Le guide énumère trois catégories proposées de logiciels, ceux de niveau 1 étant les plus importants pour la sûreté et ceux de niveau 3 étant les moins importants. Ainsi, les instructions détaillées données dans le guide peuvent répondre à des besoins différents selon les catégories de logiciels. Les logiciels sans importance pour la sûreté ne sont pas visés.

1. Exigences techniques

Pendant le développement d'un logiciel, il faut franchir trois étapes cruciales pour en démontrer la nature complète, correcte et sûre, à savoir :

- description des exigences du logiciel
- inspection systématique de la conception et de la mise en oeuvre du logiciel
- essai du logiciel.

Une description détaillée de ces trois étapes est donnée dans la présente section. Une autre étape importante, le processus et la gestion du processus employé pour élaborer le logiciel, est examinée à la section 2.

1.1 Exigences du logiciel

Les caractéristiques auxquelles doit correspondre la description des exigences (2) sont décrites ci-dessous et, sauf indication contraire, s'appliquent aux logiciels de niveaux 1, 2 et 3. Il n'est pas dans le propos du présent guide de donner des instructions sur la manière de satisfaire à ces caractéristiques, car cette question relève plutôt des normes de développement du logiciel et des procédures proposées par le titulaire du permis.

Le document décrivant les exigences du logiciel doit être clair et cohérent. Il doit être organisé logiquement de manière à permettre aux personnes qui l'examinent et à la CCEA de vérifier systématiquement s'il est précis, exact, cohérent et complet. On doit réserver à chaque point d'information une seule place précise et identifiable dans le document. Les caractéristiques doivent être indiquées de telle sorte que l'on puisse apporter des changements ultérieurement sans introduire d'incohérences.

Les exigences du logiciel doit être conforme au rôle que joue le logiciel dans l'ensemble du système et son environnement. Il doit par ailleurs décrire de façon exacte le comportement prévu lors de la mise en oeuvre du logiciel. Si le comportement effectif en cours de mise en oeuvre diffère un tant soit peu du cahier des charges, il faut corriger soit le logiciel soit le cahier des charges en conséquence.

Les exigences du logiciel doit constituer une base solide pour le travail ultérieur de développement. Il doit comporter toutes les informations voulues sur le rôle du logiciel dans le système et son environnement, en particulier sur les interfaces avec l'utilisateur, les autres systèmes et les composantes. Il doit prévoir toutes les situations normales et toutes les situations anormales vraisemblables qui sont susceptibles de se présenter pendant l'exploitation de l'ordinateur et du système fonctionnel. Plus particulièrement, le cahier des charges doit prévoir les exigences fonctionnelles, le rendement, la sûreté, la fiabilité, l'entretien et l'interface utilisateur. Il doit être assez complet pour que, lorsque toutes les exigences sont satisfaites, le logiciel soit jugé adéquat et acceptable.

LES CARACTÉRISTIQUES FONCTIONNELLES décrivent tous les entrées et sorties et le rapport fonctionnel entre les entrées et les sorties. À moins qu'elle ne soit déjà comprise intégralement dans un document de conception du système, la description des entrées et des sorties doit établir les rapports précis entre les quantités relevées dans le milieu et les entrées du logiciel, et entre les sorties du logiciel et les quantités contrôlées dans le milieu. Il importe tout particulièrement d'établir comment des quantités continues et illimitées pourront être représentées par des données limitées et distinctes. Il faut couvrir tout le champ des valeurs des entrées. Pour un logiciel de niveau 1, on doit employer une notation formelle. Pour un logiciel de niveau 2, on doit employer une notation formelle, des techniques schématiques, un pseudocode, le langage structuré ou une combinaison de ces moyens (le langage naturel ne doit pas être utilisé); le langage naturel est réservé au logiciel de niveau 3.

LES EXIGENCES DE RENDEMENT incluent tous les rapports de temps entre les entrées et les sorties, y compris les écarts permis dans le temps. Les exigences de rendement devraient également comporter des exigences concernant les ressources.

LES EXIGENCES DE SÛRETÉ précisent le comportement sûr du logiciel ainsi que les contraintes pesant sur ce comportement, en égard à l'analyse de risque du niveau du

système. Si l'évitement ou l'atténuation des risques liés au système dépend d'une exigence fonctionnelle ou de rendement donnée, il faut mentionner cette exigence.

LES EXIGENCES DE FIABILITÉ établissent les objectifs numériques de fiabilité du logiciel, en fonction d'objectifs de fiabilité au niveau du système et d'objectifs de fiabilité ou de la fiabilité connue d'autres composantes. Il faut donner une définition de ce qui constitue une défaillance, aux fins de la fiabilité. Consulter la section 1.3, qui contient des informations sur l'essai aléatoire servant à montrer que l'objectif de fiabilité d'un logiciel de niveau 1 est atteint. Le logiciel de niveau 2 ou 3 comporte des objectifs de fiabilité qui n'ont pas à être démontrés explicitement.

LES EXIGENCES DE MAINTENANCE précisent les fonctions ou les interfaces qui sont censées changer au cours de la vie opérationnelle du logiciel, notamment les interfaces avec les autres systèmes, l'utilisateur ou les composantes matérielles, et doivent préciser les exigences concernant les inspections et les essais périodiques. Si possible, ces exigences doivent également indiquer la fréquence des changements, le nombre de changements et le calendrier correspondant des changements prévus.

LES EXIGENCES CONCERNANT L'INTERFACE UTILISATEUR doivent être clairement indiquées, afin que le logiciel appuie adéquatement les fonctions et les tâches de l'utilisateur. Les exigences des utilisateurs, pour la gamme prévue de conditions d'exploitation de l'installation, devraient être puisées dans les résultats de l'analyse des fonctions et des tâches. Le lecteur trouvera dans la bibliographie (14) d'autres informations sur l'analyse des fonctions et des tâches.

Les énoncés décrivant le logiciel doivent être clairs et sans ambiguïté, afin que nul doute ne subsiste quant à leur sens. Quand une caractéristique permet plusieurs comportements, il faut préciser le comportement autorisé et celui qui ne l'est pas. Chaque exigence doit être suffisamment précise pour que l'on puisse, à l'aide d'un essai ou d'une vérification, distinguer entre une mise en oeuvre correcte ou incorrecte et, ainsi, élaborer des critères d'acceptation de cette mise en oeuvre.

1.2 Inspection systématique de la conception et de la mise en oeuvre du logiciel

Il faut effectuer une inspection systématique de la fonctionnalité et de la sûreté du logiciel afin d'obtenir l'assurance que le logiciel fait ce qu'il doit faire et qui est sûr et qu'il ne fait rien qu'il ne doit pas faire ou qu'il n'est pas dangereux. Partant du principe que les méthodes de conception et de mise en oeuvre des logiciels font l'objet de changements profonds, le présent guide ne prescrit l'emploi d'aucune méthode d'inspection particulière. Cependant, les produits des étapes de l'élaboration du logiciel devraient satisfaire aux critères « complet, correct et sûr » (prendre connaissance d'autres attributs de qualité à la section 2.3). En outre, la vérification et

la validation mentionnées à la section 1.2 s'appliquent aux logiciels et non aux facteurs humains, lesquels ne sont pas visés par le présent guide.

ANALYSE FONCTIONNELLE - Sauf indication contraire, les dispositions suivantes s'appliquent aux logiciels de niveaux 1, 2 et 3. L'analyse fonctionnelle doit :

- montrer que le logiciel accomplit toutes les fonctions prévues et n'effectue pas de fonctions non prévues; pour un logiciel de niveau 1, la CCEA recommande que la description des exigences du logiciel et que la documentation sur la conception soient fondées sur une notation et des techniques formelles, afin que l'analyse fonctionnelle puisse employer des méthodes mathématiques et des outils informatisés
- vérifier que chaque produit est complet, cohérent à l'interne et cohérent avec les produits précédents, et qu'il se conforme aux normes et aux guides précisés dans le plan du projet
- valider le système en confirmant qu'il correspond au cahier des charges et aux exigences précisées concernant l'interface utilisateur. La validation diffère de la vérification en ce qu'elle compare le produit fini à l'intention de départ, tandis que la vérification contrôle le produit de chaque phase de développement par rapport au produit de la phase précédente.

ANALYSE DE LA SÛRETÉ DU LOGICIEL - L'analyse de la sûreté et des risques du système, employée pour classer la criticalité du logiciel devrait être présentée à la CCEA aux premiers stades du projet, afin que soit établi à ce moment le niveau des analyses à suivre.

Il faut intégrer à la conception du logiciel des caractéristiques de sûreté intégrée et de tolérance aux défaillances, là où l'accroissement de la sûreté justifie une complexité supérieure. L'analyse de la sûreté du logiciel doit comporter les opérations prévues et les opérations possibles et connues de l'interface utilisateur.

L'analyse de la sûreté du logiciel devrait montrer que le logiciel ne favorise pas d'états dangereux du système, dans des conditions prévues d'exploitation ou en cas d'accident. Une analyse des risques au niveau du système doit permettre de relever les risques liés au système et de remonter à leur origine, afin que l'on puisse établir la contribution du logiciel à chaque risque. Il faut revoir et perfectionner l'analyse au fur et à mesure de la mise en oeuvre.

Si une défaillance du logiciel risque d'avoir des conséquences graves, l'analyse des dangers doit être plus poussée et porter sur la conception et le code du logiciel. Cette analyse peut entraîner des changements qui réduisent la fréquence des états dangereux ou des changements permettant d'atténuer les conséquences de certaines défaillances, afin qu'elle ne dégénèrent pas en états dangereux.

Le logiciel de niveau 1 doit être assez simple pour rendre faisable et plausible une analyse complète du logiciel. Les logiciels critiques pour la sûreté doivent être isolés des logiciels non critiques. Une isolation physique, par composantes matérielles, est préférable à une simple isolation logicielle.

1.3 Essai du logiciel

On met un logiciel à l'essai pour déceler et éliminer les défauts et établir la confiance dans la sûreté et la fiabilité du produit fini.

L'essai fonctionnel porte sur le logiciel et comporte des entrées visant la fonctionnalité et la logique du logiciel.

L'essai aléatoire porte sur le système et comporte des entrées choisies au hasard dans un profil d'exploitation réaliste.

PLAN DE L'ESSAI - Le plan de l'essai doit être rédigé avant le début de l'essai. Il doit établir les critères de l'étendue de l'essai. Il faut tenir un registre de toutes les procédures d'essai et des résultats. Des renvois doivent être établis entre les résultats de l'essai et le plan de l'essai, la configuration du matériel d'expérimentation et des composantes mises à l'essai et le registre de contrôle des changements issus de problèmes décelés pendant l'essai.

ESSAI FONCTIONNEL - Pour un logiciel de niveau 1 ou 2, l'essai fonctionnel comporte l'essai d'intégration, l'essai du système et l'essai unitaire. Pour un logiciel de niveau 3, l'essai fonctionnel comporte l'essai d'intégration et l'essai du système (il n'est pas nécessaire de procéder à l'essai unitaire, étant donné que certaines méthodes d'élaboration de logiciels remplacent cet essai par une vérification formelle, mais un essai unitaire peut être intégré à la méthode d'élaboration).

L'essai fonctionnel vérifie chaque fonction du logiciel et toutes les défaillances que peut tolérer ou atténuer le logiciel. Il faut enregistrer une mesure du champ de l'essai, y compris les cas d'espèce, les cas de valeur limite et les cas représentant tous les dangers du système. L'essai fonctionnel comporte également une vérification des exigences de séquençement et de rendement du logiciel exploité sur l'ordinateur cible.

ESSAI ALÉATOIRE - L'essai aléatoire s'applique exclusivement au logiciel de niveau 1. La CCEA tient compte du fait que des recherches sont en cours dans le domaine de la fiabilité des logiciels et que certaines hypothèses employées pour appliquer des modèles de fiabilité à des modèles sont sujettes à caution. Cependant, un essai aléatoire statistiquement valable confère une certaine assurance quant à la capacité du produit de fonctionner sans défaillance, dans certaines conditions d'exploitation, et montre que l'objectif de fiabilité est atteint.

Dans le cas d'un logiciel de niveau 1, on doit prouver la validité statistique en montrant qu'un grand nombre d'essais indépendants et aléatoires ont été effectués sans défaillance. Le nombre d'essais à effectuer est déterminé par le modèle de fiabilité employé. Il faudrait prouver que la sélection de données d'entrée employées pour les essais aléatoires représente de façon exacte soit les conditions réelles d'exploitation, soit les conditions d'exploitation dans le domaine le plus préoccupant. Enregistrer une mesure du champ des essais et faire le rapprochement entre le nombre d'essais et les exigences de fiabilité. Choisir une méthode fiable pour déceler les défaillances et faire porter des essais sur le système qui sera installé ou un équivalent proche.

2. Processus de développement des logiciels, gestion et assurance de la qualité

2.1 Plans, normes et procédures

Au début du processus de développement d'un nouveau logiciel, dans le cadre du propos du présent guide, le titulaire du permis doit présenter à la CCEA une analyse de l'importance du système pour la sûreté et indiquer à l'organisme tout autre classification particulière du logiciel. Les résultats de cette analyse détermineront les autres documents à présenter.

Tout processus de développement d'un logiciel doit suivre un plan de développement et un plan d'assurance de la qualité. Dans le cas d'un logiciel de niveau 1 ou 2, les plans doivent être présentés à la CCEA dès le début du processus; dans le cas d'un logiciel de niveau 3, il suffit que ces plans soient disponibles pour inspection.

Lorsqu'il convient d'apporter des changements à un logiciel existant, les plans de maintenance et d'assurance de la qualité du logiciel initial, et qui ont été approuvés doivent faire l'objet d'un suivi à la CCEA. Les plans de maintenance du logiciel doivent être mis à la disposition de la CCEA.

Les plans de développement et les plans de maintenance du logiciel doivent mentionner les normes dominantes et les procédures à suivre. La présente section donne des instructions quant au contenu des normes et des procédures. Le plan à proprement parler définit ou renvoie à d'autres documents qui définissent :

- le processus à suivre
- les documents à produire
- les rôles, les responsabilités et les compétences requises des employés
- la structure de l'organisation
- les installations et les outils à employer.

Dans le cas de logiciels de niveaux 1 et 2, une liste des documents à produire facilite la planification des évaluations faites par les employés de la CCEA.

2.2 Gestion de la configuration

La gestion de la configuration du logiciel devrait faire partie du système global de gestion de la configuration de la centrale; toutefois, les logiciels et leur documentation connexe doivent faire l'objet d'une attention particulière en raison du caractère changeant et invisible des données électroniques. La gestion de la configuration des

logiciels fondée sur une norme reconnue, par ex., la norme IEEE 828(5), fait partie intégrante des manuels et procédures d'un programme d'assurance de la qualité et tient compte de l'organisation des développeurs, des examinateurs et des exploitants. Il faudrait employer des outils informatisés pour faciliter et faire employer les procédures de gestion de la configuration.

Il faudrait attribuer aux logiciels un code d'identification exclusif et établir la liste des composantes identifiées exclusivement et des documents s'y rapportant. La gestion de la configuration détermine :

- comment et quand ces identificateurs exclusifs sont attribués
- comment sont contrôlés les changements apportés aux articles identifiés
- comment sont maintenus les rapports entre les articles identifiés, à mesure que des changements sont apportés
- la personne responsable de chaque aspect.

Les documents présentés à la CCEA devraient être ceux qui sont utilisés par le titulaire du permis; des documents ne devraient pas être produits simplement pour satisfaire aux exigences de la réglementation.

2.3 Processus de développement des logiciels

Le titulaire du permis doit suivre des normes et des procédures reconnues sur la manière dont les logiciels doivent être développés. La CCEA a accepté des normes et procédures industrielles à utiliser pour des projets particuliers. Comme ces normes et procédures sont encore en cours d'élaboration, il faut obtenir des avis au sujet de leur acceptabilité. À long terme, les normes et les procédures relatives au développement de logiciels s'établiront et, lorsqu'elles auront été acceptées par la CCEA, il ne sera plus nécessaire de les réévaluer chaque fois que l'on voudra s'en servir.

Les normes et les procédures doivent être fondées sur des principes de développement du logiciel bien connus et bien compris ainsi que sur l'expérience acquise dans le cadre de projets similaires de développement de logiciels et de systèmes. Il faut prévoir un mécanisme permettant d'évaluer l'expérience acquise pendant chaque projet et intégrer les améliorations aux normes et aux procédures. Le développement de logiciels devrait faire l'objet d'une assurance de la qualité au moins aussi poussée que les autres systèmes jouant dans la centrale des rôles similaires. Le titulaire du permis établit un plan de projet pour le développement de logiciels et le met constamment à jour au fur et à mesure que la situation évolue. Les normes, les procédures et les plans devraient couvrir les sujets visés à la section 2.3.

CYCLE DE VIE DES LOGICIELS - Le cycle de vie d'un logiciel décrit la période qui débute à la conception d'un logiciel et se termine lorsque le produit n'est plus disponible (9). Cette période doit être divisée en une série planifiée et contrôlée d'étapes à déroulement logique, à savoir : établissement des exigences, conception, mise en oeuvre, intégration, installation, exploitation, maintenance, succession. Chacune de ces phases doit permettre des itérations, des vérifications, des essais et des remaniements.

MÉTHODE - La méthode employée pour concevoir et documenter le logiciel devrait tenir compte de la facilité d'examen. Une spécification formelle réduit l'ambiguïté et permet une vérification formelle des systèmes simples; elle est donc recommandée pour le logiciel de niveau 1. Elle peut également être utile pour la spécification d'un logiciel de niveau 2, pour lequel, toutefois, la vérification formelle ne pourrait pas être réalisable. Le logiciel de niveau 3 peut être spécifié et vérifié de manière informelle, l'accent étant mis sur la compréhensibilité et la clarté.

ATTRIBUTS DE LA QUALITÉ DE CONCEPTION - Les normes utilisées devraient établir les attributs de la qualité de conception à réaliser; les plans et les procédures devraient mettre l'emphase sur la manière dont ces attributs seront réalisés. Les informations techniques contenues dans le présent guide laissent entendre que la conception doit être complète, correcte, simple, prévisible, robuste, cohérente, structurée, vérifiable, modifiable, retraçable, modulaire et compréhensible (voir le glossaire à la page 18).

DOCUMENTATION - Chaque activité du cycle de vie devrait aboutir à un produit (habituellement un document) qui démontre concrètement que l'activité est achevée. Un seul et même ensemble de documents doit être utilisé par toutes les personnes chargées de développer, d'examiner et de maintenir le logiciel. Il faut présenter à la CCEA les documents qui sont utilisés pour la réalisation du projet, lesquels doivent être précis, exacts, cohérents, complets, vérifiables, modifiables et retraçables (voir le glossaire à la page 18).

OUTILS - Les outils à utiliser pour développer le logiciel devraient être choisis ou élaborés en fonction de critères explicites découlant des impératifs du projet préalablement définis. Il faudrait produire un rapport d'évaluation faisant état de la conformité de chaque outil à ces critères. Il faut mettre en vigueur et en application des politiques et des procédures afin de garantir l'utilisation appropriée des outils.

MAINTENANCE - La maintenance du logiciel est le processus consistant à modifier le logiciel en fonction du changement des exigences ou d'erreurs de conception ou de mise en oeuvre découvertes pendant l'exploitation. Lorsque des problèmes, des symptômes de défaillance ou des changements aux exigences sont identifiées, il faut procéder à une analyse visant à établir la meilleure manière de corriger cette situation et ses effets sur la sûreté. Avant d'apporter des changements à un logiciel, il faut

préparer un plan de maintenance qui doit comporter des processus de développement, d'inspection et d'essai au moins aussi complets que pour le produit initial.

GESTION DU PROJET - Les responsabilités de gestion (6) (8) devraient comporter les tâches suivantes :

- établir une politique et une culture de la sûreté
- planifier, établir et mettre en oeuvre un programme d'assurance de la qualité et un logiciel efficace de sûreté
- planifier les activités du cycle de vie, ce qui comprend des jalons, des horaires et la production de documents à être présentés à l'organisme de réglementation
- établir des voies de communications appropriées entre les participants et notamment les décideurs, les experts en systèmes, le personnel d'exploitation et les experts en logiciels
- intégrer le développement du logiciel au l'ingénierie des systèmes
- établir les compétences requises pour chaque tâche et faire en sorte que seuls des employés compétents soient affectés à ces tâches (voir la section 2.3)
- établir la responsabilité, les obligations de rendre compte et l'autorité touchant au logiciel et aux documents
- fournir un soutien technique et un environnement de travail convenable
- exercer le contrôle des sous-traitants (voir la section 2.3)
- améliorer le processus.

INDÉPENDANCE - Chaque produit doit être examiné par au moins une personne indépendante de l'équipe de développement (et sans compter l'évaluation par la CCEA). Une inspection et une analyse systématiques de la fonctionnalité et de la sûreté (voir la section 1.2) doivent être effectuées et gérées par des personnes indépendantes de celles qui sont chargées de la conception et de la mise en oeuvre des logiciels. À l'exception possible des essais unitaires, la spécification des essais et les essais à proprement parler devraient être effectués par des personnes indépendantes des développeurs. Il faudrait prévoir des dispositions pour que le processus fasse l'objet d'audits indépendants, afin de vérifier que les plans et les procédures d'assurance de la qualité sont suivis et de relever les éventuelles améliorations à apporter au processus. Par indépendantes, on entend d'autres personnes utilisant des méthodes différentes, relevant d'un autre gestionnaire et utilisant un autre budget. Les logiciels de niveau 1 devraient faire l'objet du plus haut degré d'inspection indépendante. Dans le cas des logiciels de niveaux 2 et 3, la séparation gestionnelle et

budgétaire peut être moins stricte, mais il reste que les personnes affectées à la vérification du produit ne doivent pas être celles qui ont travaillé à son élaboration.

COMPÉTENCES DES EMPLOYÉS - Le titulaire du permis doit être prêt à prouver que la personne ou l'équipe affectée à la création et à l'examen d'un produit possède les compétences requises pour ce faire (6). Le plan de développement du logiciel devrait indiquer les compétences requises pour chacun des rôles du projet. Un programme de formation établi garantit que les employés acquièrent et conservent les compétences appropriées pour le rôle qui leur est confié.

TAILLE DE L'ÉQUIPE - Les équipes chargées du développement, de l'inspection et de l'essai doivent être adéquates à la taille et à l'envergure du projet. Les rôles et les responsabilités de chaque membre d'équipe devraient être définis clairement dans les plans de projet. Étant donné qu'un logiciel de niveau 1 est censé être de conception simple, une équipe nombreuse pourrait révéler une complexité excessive susceptible d'engendrer des erreurs liées à de mauvaises communications. Un logiciel de niveau 2 ou 3 peut être plus compliqué, et sa conception peut nécessiter une équipe plus nombreuse. Peu importe la taille de l'équipe, de bonnes communications et la séparation des responsabilités sont des critères importants.

CONTRÔLE DES SOUS-TRAITANTS - Lorsqu'une partie du développement, de l'inspection systématique ou des essais est confiée à des sous-traitants, il incombe au titulaire du permis de veiller à ce que les sous-traitants observent les modalités et les conditions qui s'appliquent au reste du projet. Le titulaire de permis doit répondre de tous les documents présentés à la CCEA par lui ou en son nom.

2.4 Emploi de logiciels déjà développés

Le titulaire de permis qui obtient un logiciel d'une autre organisation a la responsabilité de s'assurer que ce logiciel a été développé, inspecté et soumis à des essais conformément aux normes applicables. Dans le cas d'un logiciel qui a été développé pour une utilisation plus large que l'application à laquelle le destine le titulaire du permis, des correctifs peuvent être acceptés, comme on peut le lire à cette section.

SPÉCIFICATIONS DES EXIGENCES - Les spécifications du produit peuvent être acceptées, dans la mesure où elles décrivent de manière complète et sans ambiguïté le logiciel, d'un point de vue externe. Les spécifications des exigences doivent indiquer les contraintes exercées sur l'environnement de travail et les réponses aux entrées illégales.

INSPECTION SYSTÉMATIQUE - Il doit y avoir évidence que la conception et la mise en oeuvre sont correctes et complètes, en égard aux spécifications du produit. Le

titulaire du permis devrait montrer que les spécifications décrivent un produit qui répond aux besoins définis par la conception du système. Le titulaire du permis devrait effectuer une analyse de la sûreté afin de montrer que le logiciel ne peut contribuer à des dangers connus au niveau du système.

ESSAIS - Le titulaire de permis doit planifier et effectuer des essais d'acceptation conçus en fonction des spécifications du produit et faire rapport à leur sujet. Ces essais doivent porter sur toutes les fonctions requises et sur les conditions limites, surtout lorsque l'application visée risque de menacer le champ de validité du produit. Il faut également effectuer des essais simulant les conditions d'exploitation prévues. Des données sur l'utilisation du produit, si adéquates, peuvent être acceptées pour établir la confiance en la fiabilité du produit et peuvent remplacer les essais aléatoires. Dans un tel cas, il faut justifier qu'un processus méticuleux et complet de collecte de données et de rapports est utilisé. Il faudrait alors justifier que les autres utilisations du produit se trouvent dans des environnements et des applications semblables à ceux qui sont proposés.

PROCESSUS DE DÉVELOPPEMENT - Il n'est pas nécessaire de présenter des plans, des normes et des procédures d'avance si le développement du produit est achevé. Toutefois, l'on doit pouvoir justifier que des normes organisationnelles, des normes de développement de logiciels et de bonnes pratiques de gestion existaient au moment où le produit a été développé.

Les logiciels déjà développés qui sont destinés à faire partie d'un nouveau système ou d'une nouvelle composante devraient être évalués au même titre qu'un nouveau logiciel. Les logiciels déjà développés destinés simplement à servir d'outil de développement de logiciels peuvent être évalués d'une façon moins rigoureuse, à condition que les sorties utilisées de l'outil informatique fassent l'objet de vérifications manuelles appropriées.

3. Considérations relatives au système

Pour constituer un système, les logiciels doivent inévitablement être intégrés aux autres composantes, y compris le matériel et les utilisateurs. Le présent guide n'a pas comme propos le développement du matériel ni du système complet et ne traite pas de l'aspect des facteurs humains. Cependant, il existe certains aspects touchant au système et au matériel qui sont intimement liés au logiciel. La présente partie donne des instructions au sujet de l'évaluation des interfaces entre le logiciel et le reste du système.

3.1 Spécification du système

Un système est une entité matérielle limitée qui accomplit, dans son environnement, un travail défini par l'interaction de ses parties. Le développement du logiciel part du

principe que les limites et l'objet du système ont été définis dans la spécification du système. Si la fonctionnalité du système est principalement accomplie par le logiciel et si la spécification du système répond aux conditions décrites à la section 1.1, il peut ne pas être nécessaire de produire une spécification distincte des exigences pour le logiciel. Dans un tel cas, la spécification du système peut être évaluée comme les exigences du logiciel. Toutefois, il est également acceptable que la spécification du système décrive les exigences du système de manière moins détaillée lorsqu'une spécification distincte des exigences du logiciel fournit une description complète, formelle et détaillée du logiciel. Dans ce cas, il importe de pouvoir retracer et vérifier chaque exigence de la spécification distincte des exigences du logiciel, en fonction de la spécification du système. Dans un cas comme dans l'autre, la validation finale de la nature correcte du système doit être faite en regard de la spécification du système.

3.2 Conception du système

La conception du système devrait indiquer les rôles du matériel et du logiciel dans le système. Ces rôles et les interfaces entre ces composantes doivent être indiqués de manière complète et sans ambiguïté. Pour comprendre clairement comment le logiciel fonctionnera, la spécification du matériel doit définir complètement le comportement externe de toutes les instructions logicielles légales et décrire les mécanismes de traitement des erreurs. Il est prévu que la sélection du matériel informatique suppose la compréhension du rôle du logiciel et que des itérations de la conception du système peuvent être requises à mesure que les rôles seront précisés.

Afin d'atteindre les exigences du système en matière de sûreté et de fiabilité, il peut être nécessaire de concevoir le système de manière qu'il puisse affecter plusieurs composantes différentes à une fonction ou à des fonctions similaires. Par exemple, les documents de réglementation R-8 et R-10 de la CCEA exigent deux systèmes d'arrêt rencontrant l'exigence de diversité et indépendants dans les réacteurs nucléaires canadiens (10) (11). Il faut admettre qu'il y a risque pour la diversité de la conception lorsque plusieurs composantes utilisent un logiciel pour une fonctionnalité identique ou similaire. Il faudrait que la conception tienne compte de ce danger en prévoyant d'autres types de diversité comme la diversité fonctionnelle, des capteurs indépendants et rencontrant l'exigence de diversité en temps de réponse. Les limites pratiques actuelles de la quantification de la fiabilité des logiciels portent à conclure qu'aucune composante de logiciel ne devrait avoir un taux d'indisponibilité supérieur à environ 10^{-4} . Si l'on souhaite obtenir une fiabilité supérieure pour le système, il faut employer des composantes logicielles multiples et rencontrant les exigences de diversité.

3.3 Interface matériel-logiciel

Plusieurs interfaces matériel-logiciel influenceront sur le développement et l'analyse du logiciel. L'analyse fonctionnelle, la conception et le codage du logiciel dépendent d'une définition complète et non ambiguë du jeu d'instructions informatiques. Cette interface peut être placée dans un compilateur d'un langage de programmation supérieur, dans le système d'exploitation ou dans les pilotes de périphériques, pour les pièces et les composantes de matériel périphérique. L'analyse et l'essai de temps de réponse dépendent du temps de réponse, du séquençement et des interruptions du matériel. L'analyse de temps de réponse étant généralement ardue et sujette à erreur, elle devrait être suivie de l'essai de rendement. L'essai de rendement devrait être effectué sur la configuration matérielle cible et ne devrait pas être intrusif (c'est-à-dire que l'environnement d'essai ne devrait pas ajouter du matériel ou du logiciel susceptible de modifier le rendement).

L'analyse de la sûreté du logiciel devrait faire partie d'une analyse globale de la sûreté du système, dans le cas des systèmes de niveau 1. L'analyse de sûreté du logiciel repose sur la connaissance des modes, des effets et des fréquences des défaillances du matériel. Une analyse de défaillance du matériel peut entraîner pour le logiciel la nécessité de surveiller le matériel et de réagir à des défaillances détectées.

ASSEMBLEURS, COMPILATEURS ET AUTRES OUTILS GÉNÉRATEURS DE CODES - Le développement de logiciels comporte toujours l'emploi d'outils destinés à traduire une conception de logiciel, exprimée en langage de programmation, en un code pouvant être exécuté par le matériel informatique. Parmi les langages de programmation, on note des langages de conception de haut niveau et des langages assembleurs détaillé de bas niveau. Les outils qui traduisent des langages de haut niveau contiennent des pièces de logiciel qu'ils incorporent au produit; c'est pourquoi il faut les considérer comme des logiciels pré-développés. D'un autre point de vue, ils sont des outils de conception et c'est leur produit (le code exécutable) qui doit faire l'objet d'un examen détaillé.

Les outils de conception doivent être choisis et évalués en regard de critères établis (voir la section 2.3). Le fournisseur de logiciels pré-développés doit pouvoir justifier d'un bon processus d'assurance de qualité et de contrôle du produit et faire la preuve de vérifications et d'essais approfondis (voir la section 2.4). La certification ou l'approbation d'un outil par un organisme international ou national de validation est un atout mais ne peut être agréé sans question. Les développeurs d'applications de niveaux 1 et 2 devraient examiner le langage de programmation et le traducteur et choisir un sous-ensemble du langage et des caractéristiques de l'outil qui ne contiennent que des parties bien définies, prévisibles et bien comprises. Cela devrait être décrit dans les procédures de conception et de codage du logiciel.

Les assembleurs, les compilateurs et les autres outils générateurs de codes comportent généralement une vérification automatique de la conception du logiciel,

pendant la traduction. Une telle vérification est souhaitable et peut être incorporée au rapport de vérification. Ces outils peuvent également générer des vérifications à l'exécution. Si c'est le cas, il faut veiller à les inclure explicitement au processus d'autovérification.

L'essai de logiciel est généralement la seule vérification qui est faite des produits de ces outils. Chaque essai effectué sur le code exécutable vérifie également la nature correcte des outils. Si les résultats de l'essai révèlent un écart avec les prévisions, une erreur de l'outil est une possibilité, auquel cas il faut la corriger. L'essai le plus direct de l'outil qui crée le code exécutable consiste à renverser de manière indépendante le processus et à reproduire la conception détaillée à partir du code exécutable. Cela n'est habituellement pas faisable mais il peut parfois être nécessaire, dans des circonstances particulières, de fournir une confiance additionnelle en la partie la plus cruciale du logiciel, ou encore dans le cas d'un logiciel que l'on ne peut essayer directement.

SYSTÈMES D'EXPLOITATION - Les systèmes d'exploitation devraient être traités comme des logiciels pré-développés. Il ne faut pas utiliser des systèmes d'exploitation complexes pour les logiciels de niveau 1. Un petit noyau du système d'exploitation peut être utilisé si l'on peut l'assurer au même niveau que le logiciel d'application; toutefois, le cas échéant, il faut retirer toutes les routines non essentielles, afin d'éviter des effets secondaires inattendus et non souhaités. Les systèmes d'exploitation posent des problèmes particuliers touchant à la maintenance des logiciels. La maintenance du fournisseur du système d'exploitation pourrait ne pas respecter les exigences de l'application particulière d'un titulaire de permis, et d'autre part, il pourrait être difficile d'obtenir les droits, la documentation et les compétences voulus pour prendre en charge et adapter la maintenance. Ces questions devraient être examinées et des réponses devraient être fournies dans le plan de développement du logiciel.

TEMPS DE RÉPONSE ET RENDEMENT - L'on sait que les exigences en matière de temps de réponse et de rendement sont difficiles à préciser et à vérifier, en général. Pour les systèmes de niveau 1, le matériel et les logiciels devraient être conçus de sorte que le temps de réponse est strictement contrôlée et prévisible. Dans le cas des systèmes de niveaux 2 et 3, l'on peut employer des conceptions plus souples, qui, dans certains cas, peuvent devenir nécessaires. Afin de pouvoir prévoir le temps de réponse et le rendement dans un environnement informatique donné qui comporte plusieurs processus interruptibles, chaque processus doit être spécifié, conçu et mis en oeuvre de manière modulaire, afin que les besoins et les effets du temps de réponse et des processus soient explicitement définis et que les interactions entre les processus soient comprises.

AUTO-VÉRIFICATION - L'auto-vérification est la capacité d'un système de déceler des défaillances de composantes et d'intervenir. Les systèmes qui comportent des logiciels

peuvent être conçus pour déceler quelques défaillances de capteurs, de dispositifs ou de matériels et même pour reconnaître des défaillances d'autres parties de logiciels. Après avoir décelé de telles défaillances, le logiciel produit des sorties avec sûreté intégrée, tolérante aux pannes ou de diagnostic. L'auto-vérification offre certains avantages qui ont toutefois comme prix d'accroître la complexité du système.

Il faut procéder à une analyse des risques du système ou analyse des modes de défaillance et de leurs effets pour identifier les défaillances matérielles à relever. À l'issue de cette analyse, la liste des défaillances matérielles qui peuvent et doivent être détectées est incorporée à la spécification des exigences du logiciel. À défaut d'effectuer cette analyse (par exemple dans le cas de systèmes de niveaux 2 et 3), l'on peut fonder les exigences d'auto-vérification sur l'expérience acquise avec des systèmes et des composantes similaires.

Dans le cas de défaillances logicielles, les erreurs courantes comme le débordement de piles ou la division par zéro sont faciles à déceler. Les mécanismes de détection doivent être décrits dans une procédure de codage propre au projet qui indique également les réponses appropriées. Le logiciel a plus de mal à déceler les défaillances logicielles plus subtiles et plus compliquées, pendant l'exploitation, de sorte qu'une telle auto-vérification n'est généralement pas recommandée. La conception de systèmes de niveaux supérieurs devrait tenir compte de la possibilité de défaillance de composantes logicielles.

Glossaire

Note : Les définitions suivies d'un numéro renvoient au document correspondant à la section des références de ce guide, dont elles sont tirées ou adaptées.

ANALYSE DES TÂCHES :	Évaluation systématique des fonctions attribuées aux employés de la centrale, dans le but d'identifier les exigences des utilisateurs.
ANALYSE DE LA SÛRETÉ	Démonstration que le logiciel ne contribue pas à des mesures dangereuses dans des conditions d'exploitation attendues et dans des conditions d'accident.
ANALYSE DES RISQUES :	Les risques du système désignent des conditions ou des états du système qui pourraient conduire à des accidents. L'analyse des risques est l'examen d'un système du point de vue de la sûreté et consiste à identifier et à évaluer les risques éventuels dans le but de les éliminer ou de les réduire, ou du moins d'atténuer leurs effets.
ANALYSE FONCTIONNELLE :	Preuve que le logiciel accomplit toutes les fonctions voulues et n'effectue pas de fonctions non désirées. Elle comprend la vérification et la validation
ANALYSE DU SYSTÈME :	Identification des fonctions qui doivent être accomplies par des personnes et l'informatique pour réaliser des objectifs de la centrale ou du système et satisfaire à des exigences de rendement. L'analyse des fonctions constitue la base de l'analyse des tâches.
assurance de la qualité :	Ordre planifié et systématique de toutes les mesures requises pour fournir une confiance adéquate de la conformité du système et des produits matériels aux exigences techniques.
AUTO-VÉRIFICATION :	Désigne un logiciel capable de déceler des défaillances de composantes et d'y réagir
COHÉRENT :	Ne contient aucune contradiction; contient une notation, une terminologie, des commentaires, une symbologie et des techniques de mise en oeuvre uniformes.
COMPILATEUR :	Outil informatique servant à traduire une conception logicielle, exprimée dans un langage de programmation, en un code pouvant être exécuté par le matériel.

COMPLET :	Décrit ou vise tous les comportements du système ou du logiciel dans toute situation susceptible de se produire pendant l'exploitation. Les situations que l'on ne considère pas susceptibles de se produire au cours de l'exploitation devraient être clairement identifiées.
COMPRÉHENSIBLE :	Peut être compris par des personnes compétentes autres que le créateur (cela suppose une conception simple, homogène et structurée).
CORRECT :	Capable de produire les sorties spécifiées lorsque sont introduits les entrées spécifiées et mesure dans laquelle la mise en oeuvre correspond à la spécification.
CRITIQUE POUR LA SÛRETÉ :	Désigne le logiciel classé le plus critique pour la sûreté.
ESSAI :	Processus consistant à faire fonctionner (exploiter ou faire tourner) un système ou une composante de système par des moyens manuels ou informatisés, dans le but de vérifier qu'il correspond aux exigences posées ou de relever les écarts entre les résultats attendus et les résultats obtenus (12).
ESSAI FONCTIONNEL :	Essai qui tente de faire fonctionner toutes les parties d'un logiciel comme elles ont été spécifiées ou conçues, dans le but de trouver des erreurs.
MATÉRIEL :	Dans le contexte des systèmes informatiques, le matériel désigne les composantes électroniques physiques.
EXACT :	Décrit le comportement effectif du système ou du logiciel.
FORMEL :	Doté de règles strictes quant à la rédaction et à l'interprétation d'instructions; syntaxe et sémantique fondées sur des définitions mathématiques précises.
INDÉPENDANT :	Signifie qu'il s'agit de personnes différentes utilisant des méthodes et des outils différents, relevant d'un gestionnaire différent et travaillant avec un budget différent.
INSPECTION SYSTÉMATIQUE :	Analyse fonctionnelle et analyse de la sûreté du logiciel, accomplies par le titulaire du permis.

MAINTENANCE :	Modification d'un logiciel après la livraison, en vue de corriger des défaillances, d'améliorer son rendement ou d'autres attributs, ou d'adapter le produit à un changement d'environnement.
MODIFIABLE :	Doté d'une structure rigide garantissant que l'information sera présentée en un endroit clairement identifié dans le document et seulement à cet endroit, de sorte que le document puisse être modifié sans devenir incohérent.
MODULAIRE :	Mesure dans laquelle le logiciel est composé d'éléments distincts (modules), de sorte qu'un changement à un élément influe de façon minimale sur les autres éléments.
PRÉCIS :	Ne comportant pas d'ambiguïté et ne laissant aucun doute quant au sens.
PRÉVISIBLE :	Produit les sorties attendues à partir d'entrées connues; pas susceptible de produire des effets aléatoires ou impossibles à prévoir.
LOGICIEL :	Programme, procédures, règles et toute documentation connexe s'appliquant à l'exploitation d'un système informatique.
RETRAÇABLE :	Il est possible de trouver l'information initiale dans les produits des étapes précédentes du cycle de vie.
ROBUSTE :	Capable de continuer d'accomplir des fonctions établies malgré des dérogations aux hypothèses de la spécification.
SIMPLE :	Logiciel accomplissant des fonctions de la manière la plus compréhensible.
STRUCTURE :	Technique utilisée pour organiser et coder des logiciels et permettant de réduire la complexité, d'améliorer la clarté et de faciliter les modifications.
SÛRETÉ INTÉGRÉE :	Capacité d'un système de passer à un état sûr prédéterminé lorsque se présente une défaillance.
SYSTÈME :	Entité physique limitée qui accomplit, dans son environnement, une fin définie du fait de l'interaction de ses parties.

SYSTÈME D'EXPLOITATION :	Programme ou ensemble de programmes qui offre une interface entre le logiciel d'application et un ensemble donné de matériels, notamment l'initialisation, l'affectation des ressources, la séquence des processus, la gestion des fichiers et les communications.
TEMPS RÉEL :	Désigne un système ou un mode d'exploitation où le travail de l'ordinateur est effectué pendant qu'un processus externe se déroule, de sorte que les résultats du calcul peuvent être utilisés pour contrôler ou surveiller ce processus externe ou y réagir de façon opportune.(12)
TOLÉRANCE AUX PANNES :	Capacité intégrée d'un système d'assurer une exécution correcte ininterrompue en présence d'un nombre limité de défaillances matérielles ou logicielles.
VALIDATION :	Action consistant à vérifier (par un essai) qu'un système ou une description d'un système sont complets et conformes aux exigences du système et aux besoins de l'utilisateur.
VÉRIFICATION :	Processus consistant à établir si les produits d'une phase donnée du cycle de vie du logiciel correspondent aux exigences établies pendant la phase précédente et aux exigences des normes et procédures dominantes.

Références

- [1] Document de consultation C-22, *Les programmes d'assurance de la qualité des installations nucléaires*, Révision 1, Commission de contrôle de l'énergie atomique, 1991
- [2] INFO-0505, "Documentation of Computerised Safety Systems of Nuclear Power Stations", by D. L. Parnas, AECB Project 2.234.1
- [3] Document de consultation C-98, *Exigences concernant l'analyse de fiabilité des systèmes liés à la sûreté dans les réacteurs nucléaires*, Révision 1 (Ébauche 19, 95/03/20)
- [4] INFO-0247, "Computer Software Configuration Management", by G. Pelletier, PRIOR Data Science Ltd., AECB Project 2.109.1, 1987
- [5] IEEE Std 828-1990, "IEEE Standard for Software Configuration Management Plans"
- [6] CAN3-N286.2-86, *Assurance de la qualité de la conception des centrales nucléaires*, Association canadienne de normalisation, 1986
- [7] IAEA Technical Report Series No. 282, "Manual on Quality Assurance for Computer Software Related to the Safety of Nuclear Power Plants", International Atomic Energy Agency, 1988
- [8] IAEA Technical Report Series No. 367, "Software Important to Safety in Nuclear Power Plants", International Atomic Energy Agency, 1994
- [9] IEC 880, "Software for computers in the safety systems of nuclear power stations", International Electrotechnical Commission Publication 880, First edition, 1986
- [10] Document d'application de la réglementation R-8, *Exigences relatives aux systèmes d'arrêt des centrales nucléaires CANDU*, Commission de contrôle de l'énergie atomique, 1991
- [11] Document d'application de la réglementation R-10, *L'utilisation de deux systèmes d'arrêt des réacteurs*, Commission de contrôle de l'énergie atomique, 1977
- [12] IEEE Std 610.12 - 1990, "IEEE Standard Glossary of Software Engineering Terminology", Institute of Electrical and Electronics Engineers, and recognized as an American National Standard, 1990.

- [13] Committee for Review of Oversight Mechanisms for Space Shuttle Flight Software Processes, “An Assessment of Space Shuttle Flight Software Development Processes”, National Academy Press, 1993
- [14] INFO-0605, “Human Factors Guides”, PHF Services Inc, 1995, AECB project No 2.280.2