Croissance de la sûreté de fonctionnement des logiciels. Caractérisation, modélisation, évaluation

Karama Kanoun

Année 1989

THÈSE

présentée à

L'INSTITUT NATIONAL POLYTECHNIQUE DE TOULOUSE

pour obtenir

LE GRADE DE DOCTEUR D'ÉTAT

MENTION : INFORMATIQUE

par

Karama KANOUN

Ingénieur ENAC - Docteur Ingénieur

CROISSANCE DE LA SÛRETE DE FONCTIONNEMENT DES LOGICIELS CARACTÉRISATION - MODÉLISATION - ÉVALUATION

Soutenue le 21 Septembre 1989 devant le jury :

M.	J.C.	LAPRIE	Président
MM.	W.C.	CARTER	Membres
	A.	CHEILAN	
	A.	COSTES	
	G.	LE GALL	
	B.	LITTLEWOOD	
	J.G.	ROUSSEL	
	J.L.	SOLER	

Rapport LAAS n° 89.320

Cette thèse a été préparée au Laboratoire d'Automatique et d'Analyse des Systèmes du C.N.R.S.
7, avenue du Colonel Roche, 31077 TOULOUSE CEDEX

A Dorra

A Najib

"Un miroir reflète la vérité mais chacun le regarde de la place qu'il occupe"

Agatha Christie, Le train bleu.

AVANT-PROPOS

Les travaux présentés dans ce mémoire ont été effectués au LABORATOIRE D'AUTOMATIQUE ET D'ANALYSE DES SYSTEMES (LAAS) du CNRS. Je tiens à exprimer ma profonde gratitude à Monsieur G. GRATELOUP qui m'a accueillie au LAAS alors qu'il en était le directeur et à Messieurs D. ESTEVE et A. COSTES qui lui ont succédé à ce poste, pour la confiance qu'il m'ont toujours témoignée.

Je remercie tout particulièment Monsieur J.C. LAPRIE, Directeur de Recherche au CNRS, responsable du groupe "Tolérance aux fautes et Sûreté de Fonctionnement informatique" (TSF), pour m'avoir accueillie dans ce groupe. Il a su en permanence me faire profiter de son expérience et orienter mes travaux de recherche. Nos nombreuses discussions m'ont, à bien des égards, aidée dans mes recherches et dans leur aboutissement. Il a su, par son exemple, me faire découvrir et apprécier le métier de chercheur.

Je suis très reconnaissante envers Monsieur A. COSTES qui, en tant que membre du groupe TSF m'a toujours encouragée lors de la rédaction de ce mémoire. Ses critiques constructives, ses conseils pertinents ainsi que sa patience m'ont soutenue tout au long de ce travail.

Je tiens à exprimer ma profonde gratitude à J.C. LAPRIE, directeur de recherche au CNRS, pour l'honneur qu'il me fait en présidant le Jury de thèse, ainsi qu'à :

- Monsieur W.C. CARTER ancien cadre scientifique au Centre de Recherche Thomas J. Watson d'IBM, actuellement professeur à l'Université de Newcastle Upon Tyne,
- Monsieur A. CHEILAN, Ingénieur responsable de la qualité du Centre Commun de Recherches Louis Blériot de l'Aérospatiale,
- Monsieur A. COSTES, Professeur à L'Institut National Polytechnique de Toulouse, Directeur du LAAS,
- Monsieur G. LE GALL, responsable du département "Evaluation et Validation de Protocoles" du CNET à Lannion,
- Monsieur B. LITTLEWOOD, Professeur à City University à Londres, Directeur du "Centre for Software Reliability",
- Monsieur J.G. ROUSSEL, Directeur Central de la Qualité au CNES,
- Monsieur J.L. SOLER, Professeur à l'Institut National Polytechnique de Grenoble, Directeur des études de l'ENSIMAG,

pour l'honneur qu'il me font en participant à ce jury, et particulièrement à Messieurs J.C. LAPRIE, A. CHEILAN et J.L. SOLER qui ont accepté la charge d'être rapporteurs.

Les travaux théoriques présentés dans ce mémoire n'auraient pas pu être mis en pratique et étayés sans l'aide d'ALCATEL CIT, de l'Aérospatiale et de la compagnie brésilienne des télécommunications TELEBRAS CPqD. Je tiens à remercier le personnel de ces organismes de nous permettre de publier tout ou partie des résultats des études qu'ils nous ont confiées.

Je tiens à remercier tous ceux qui ont participé aux différentes études présentées dans ce mémoire : Thierry SABOURIN d'ALCATEL CIT, Jorge MOREIRA et Marta MARTINI de TELEBRAS CPqD et Mohamed KAANICHE du LAAS, pour les nombreuses discussions et fructueux échanges d'idées que j'ai eus avec eux.

Je remercie également tous les membres du groupe TSF pour leurs conseils pertinents et leur soutien amical et en particulier Jean ARLAT, Christian BEOUNES, Jean-Paul BLANQUART (actuellement à Matra Espace), Yves CROUZET, Yves DESWARTE, Jean-Charles FABRE, Daniel NOYES, Joëlle PENAVAYRE, David POWELL et Pascale THEVENOD.

Mes remerciements vont encore à tous les membres des services *Informatique*, *Réception-standard*, *Documentation-Edition* et *Direction* qui m'ont permis d'effectuer mon travail de façon efficace et agréable et qui ont facilité la réalisation de ce mémoire.

Je remercie enfin tous ceux qui, de près ou de loin, m'ont soutenue dans mon travail.

SOMMAIRE

INTRODUCTION GENERALE

Le développement de notre société est de plus en plus lié à l'informatique qui prend une place de plus en plus importante. Ceci entraîne une dépendance de plus en plus grande de l'informatique et plus particulièrement des logiciels. En effet, les logiciels sont utilisés aussi bien en tant que produit en soi (logiciel d'application : de contrôle, de commande ou de contrôle-commande) qu'en tant que support de services (aide à la production et au développement de systèmes). Il est donc indispensable de concevoir des logiciels "fiables" et de "bonne qualité" afin d'avoir confiance dans les services rendus. De plus des chiffres révèlent la prépondérance et la relative progression du coût du logiciel par rapport à celui du matériel [Boe 81] : aux USA, la part du logiciel dans le coût global du projet était de l'ordre de 70% en 1970 pour atteindre 90% en 1985.

La notion de qualité revêt trois aspects [Fer 88] :

- la qualité de production : le logiciel a été réalisé selon les règles de l'art,
- la qualité de conception : le logiciel est bien fait,
- la qualité de conformité : le logiciel rend bien le service attendu.

Ces différents aspects s'intéressent aussi bien au processus de production qu'au produit lui-même à un instant donné.

Le logiciel peut rendre le service attendu à un moment donné (qualité de conformité) tout en contenant des fautes de conception non encore activées ; l'un des objectifs de l'évaluation de la sûreté de fonctionnement d'un logiciel est de chiffrer l'impact de ces fautes de conception sur le fonctionnement du logiciel.

Les méthodes d'évaluation de la sûreté de fonctionnement doivent faire partie intégrante des méthodes et outils de l'assurance qualité afin d'obtenir le niveau de confiance.

L'évaluation de la sûreté de fonctionnement d'un logiciel permet :

- d'avoir confiance dans le service délivré,
- de disposer d'éléments concrets pour le suivi du développement,
- de s'assurer que la sûreté de fonctionnement est conforme aux objectifs exprimés dans les spécifications.

Les retombées d'une évaluation de sûreté de fonctionnement sont multiples et les retombées peuvent être aussi bien immédiates qu'à plus long terme.

- A court terme, les résultats peuvent servir dès la phase de développement du système considéré pour appuyer le contrôle de qualité.
- A moyen terme les résultats de l'évaluation permettent d'estimer la sûreté de fonctionnement du système considéré en vie opérationnelle.
- A plus long terme, les résultats de l'étude de sûreté de fonctionnement d'un système particulier, ou d'une gamme de systèmes pour un projet donné, peuvent contribuer à une meilleure conception et à une validation plus rapide de nouveaux systèmes relatifs à des projets semblables. La sûreté de fonctionnement d'un système peut ainsi être accrue au cours de son évolution dans le temps. Le processus de production est ainsi amélioré par une meilleure connaissance du produit lui-même.

Considérant les retombées à court et moyen termes, l'évaluation de la sûreté de fonctionnement d'un logiciel peut être effectuée soit pour un système donné soit pour un certain nombre de réalisations possibles pour ce système à partir des mêmes spécifications. La comparaison des résultats de l'évaluation permet d'effectuer un choix selon des critères objectifs et quantifiés parmi les différentes réalisations possibles.

L'évaluation d'un ensemble de réalisations pour un même système permet de fournir aux concepteurs des éléments de choix et de dimensionnement des différentes solutions possibles.

Dans ce mémoire nous nous intéressons à l'évaluation de la sûreté de fonctionnement d'un système informatique et plus particulièrement d'un logiciel. Nous considérons à la fois les aspects de fiabilité (prise en compte des temps hors défaillance uniquement) et de disponibilité (prise en compte simultanée des temps hors défaillance et des temps où le système est défaillant [Lap 89]). Nous nous intéressons à la sûreté de fonctionnement d'un logiciel déjà codé et qui est en cours d'évolution, cette évolution ayant pour origine soit des corrections, soit des changements de spécifications. Cette situation est couramment appelée "croissance de fiabilité".

Ce mémoire est organisé en deux parties.

La première partie est composée de deux chapitres traitant de la caractérisation et de la modélisation de la croissance de fiabilité et de disponibilité.

Dans le premier chapitre le comportement d'un logiciel en phase de croissance de fiabilité est analysé. Le domaine de défaillance est défini ainsi que les principaux facteurs influençant

soit sa nature soit son évolution. Les mesures de sûreté de fonctionnement sont ensuite définies et leur évolution en fonction du temps est étudiée.

Le deuxième chapitre est consacré à la modélisation du logiciel en phases de croissance de fiabilité et en "fiabilité" stabilisée. Dans un premier temps, nous utilisons les processus de renouvellement généralisés pour évaluer l'intensité (ou la probabilité) de défaillance et la disponibilité. Ces modèles sont appelés modèles de connaissance. L'expression de l'intensité de défaillance est établie pour différentes politiques de correction : correction immédiate ou différée ou par lot. Ces modèles de connaissance font intervenir explicitement les paramètres des différents phénomènes mis en jeu mais les expressions des mesures de sûreté de fonctionnement sont complexes. Ces modèles sont alors approchés par des modèles d'action (qui sont les modèles de croissance de fiabilité) pour lesquels les différents phénomènes mis en jeu sont pris en compte implicitement avec des expressions des mesures de sûreté de fonctionnement plus simples. Les modèles classiques de croissance de fiabilité sont tous concernés par la fiabilité au cours du développement : ils supposent que le taux de défaillance tend vers zéro quand le temps tend vers l'infini et ne tiennent pas compte des temps pendant lesquels le logiciel est défaillant. Ceci nous a amenée à développer un modèle plus adapté à la vie opérationnelle : le modèle hyperexponentiel. Outre sa capacité à tenir compte d'un taux de défaillance non nul à l'infini, ce modèle permet de modéliser la croissance de disponibilité.

Dans la deuxième partie, nous définissons et analysons une méthode globale pour l'évaluation de la fiabilité d'un logiciel. Cette méthode est constituée d'une partie qualitative étroitement liée au système étudié et d'une partie plus systématique et plus quantitative : prétraitement des données et application des modèles de croissance de fiabilité.

Au chapitre III, on s'intéresse au prétraitement des données : recherche des données douteuses et tests de tendance. Les notions de tendances locale et globale sont introduites et les tests correspondants sont développés.

Le quatrième chapitre permet de répondre aux problèmes soulevés par l'application des modèles de croissance de fiabilité : choix des grandeurs à évaluer et de la variable aléatoire à étudier, méthodes de calibration des modèles de croissance de fiabilité, application des modèles par composant et par conséquence, critères de validation de modèles... Les deux grandeurs les plus couramment utilisées pour caractériser la fiabilité d'un logiciel sont le nombre de fautes résiduelles et le taux de défaillance. Ces deux grandeurs sont étroitement liées mais ne permettent pas d'obtenir les mêmes informations. Le nombre de fautes résiduelles caractérise le logiciel de façon statique et peut servir de guide lors du développement. Le taux de défaillance quant à lui permet de tenir compte de l'environnement

d'utilisation du logiciel et est, de ce fait, plus adapté à la vie opérationnelle. Seule cette dernière quantité sera utilisée dans ce mémoire pour caractériser la "croissance" de fiabilité. Cependant, afin d'apprécier "l'amélioration" du logiciel, le suivi de l'évolution du nombre de défaillances constitue un bon indicateur du sens de variation de la fiabilité et l'évaluation du nombre de défaillances qui auront lieu durant la période suivante permet de planifier le test ainsi que la maintenance en vie opérationnelle (plus précisément le nombre de corrections qu'il faudrait continuer à apporter).

Enfin le cinquième et dernier chapitre met en œuvre la méthode d'évaluation proposée : évaluation de la fiabilité de deux autocommutateurs téléphoniques. Ces deux systèmes sont étudiés selon différents objectifs :

- évaluation de la fiabilité du logiciel relatif à un site en vie opérationnelle, pour les deux systèmes,
- suivi du développement et estimation de l'effort de maintenance à assurer en vie opérationnelle pour l'ensemble des sites installés, pour le second système.

PREMIERE PARTIE

CARACTERISATION ET MODELISATION
DE
LA CROISSANCE DE LA SURETE DE FONCTIONNEMENT

Cette première partie comporte deux chapitres :

- dans le premier chapitre nous analysons le comportement du logiciel en fonction de divers facteurs tels que la politique de correction et le domaine de défaillances, nous définissons les différentes mesures de la sûreté de fonctionnement et nous étudions l'évolution de ces mesures à spécifications inchangées et en cas de changement de spécifications,
- dans le deuxième chapitre nous établissons des modèles de comportement du logiciel (modèles de connaissance et d'action) pour différentes politiques de modification-correction.

CHAPITRE I

CARACTERISATION

DE LA CROISSANCE DE LA SURETE DE FONCTIONNEMENT

INTRODUCTION

Le comportement d'un logiciel, sa sûreté de fonctionnement à un instant donné et son évolution dans le temps dépendent de nombreux facteurs que l'on peut résumer comme suit :

- le **domaine de défaillance** du logiciel ; il s'agit essentiellement de l'ensemble des entrées dont l'activation entraîne la défaillance du logiciel,
- le **profil d'utilisation** ainsi que les conditions d'environnement,
- les modifications introduites au niveau du logiciel soit sous forme de **corrections** soit sous forme de **changement de spécifications** aussi bien en développement qu'en vie opérationnelle.

De plus le nombre de copies ou d'instances[1] en service influence la "vitesse" d'évolution du domaine de défaillance.

Ces facteurs ne sont pas indépendants et l'évolution de l'un d'entre eux influence l'évolution de toute ou partie des autres. L'étude de la "croissance" de la sûreté de fonctionnement d'un logiciel doit donc être effectuée en tenant compte de l'ensemble de ces facteurs.

Les buts de ce chapitre sont d'analyser le comportement d'un logiciel en fonction de ces différents facteurs, de définir et de caractériser les différentes mesures de la sûreté de fonctionnement et d'étudier l'évolution de ces mesures en fonction du temps.

I-1- ANALYSE DU COMPORTEMENT D'UN LOGICIEL

Le comportement d'un logiciel résulte à la fois du domaine de défaillance, du profil d'utilisation et des modifications introduites sur ce logiciel.

---

[1] Le mot copie est employé dans le sens strict du terme : copie identique à tout point de vue, alors qu'une instance correspond à l'adaptation d'un logiciel à un site particulier, avec changement de certains paramètres par exemple.

Le profil d'utilisation conditionne : (1) le choix des données d'entree dans le domaine des entrées, (2) l'échelle des temps (selon que le système est utilisé de façon continue ou discontinue) ainsi que la charge du système ; il conditionne de ce fait la vitesse d'évolution du domaine de défaillance.

Les modifications, quant à elles, influencent directement le domaine de défaillance d'un logiciel de par la nature des corrections introduites sur le logiciel et des changements de spécifications ; elles influencent aussi la vitesse d'évolution de ce domaine de par leur fréquence.

Les modifications effectuées en vie opérationnelle constituent ce qui est couramment appelé la **maintenance** du logiciel. Ces modifications sont introduites de différentes manières : juste après défaillance, à intervalles réguliers ou de façon irrégulière quand les corrections sont achevées... ; nous parlerons dans la suite de **politiques de maintenance**.

On voit que tous les facteurs influencent soit la nature soit l'évolution du domaine de défaillance ; l'analyse du comportement d'un logiciel durant le cycle de vie sera donc effectuée au travers de l'étude du domaine de défaillance et de son évolution en fonction des autres facteurs.

Afin de fixer les idées nous commencerons par définir brièvement un certain nombre de politiques de correction et de modification qui peuvent être effectuées sur un logiciel et qui seront prises en compte par la suite.
Nous définirons ensuite de façon précise le domaine de défaillance d'un logiciel et nous analyserons l'influence du profil d'utilisation et des différentes politiques de correction-modification considérées sur l'évolution de ce domaine.

I-1-1 Correction-modification du logiciel durant le cycle de vie

De l'expression des besoins à la vie opérationnelle, le logiciel passe par un certain nombre de phases [Boe 76, Leh 80, Afn 85]. L'objectif de ce paragraphe est d'analyser les différentes politiques de correction-modification tout au long de ces phases. Ces politiques sont pratiquement les mêmes pour un certain nombre de phases, ce qui nous a amenée à regrouper les différentes phases en trois étapes : l'expression des besoins, le développement et la vie opérationnelle. L'expression des besoins se traduit par le cahier des charges du système. Le développement correspond à toutes les phases allant de la spécification en passant par la conception préliminaire à la validation pré-opérationnelle. La vie opérationnelle inclut l'exploitation et la maintenance. Nous nous intéressons à la sûreté de fonctionnement d'un logiciel en cours de développement et/ou en vie opérationnelle.

En vie opérationnelle, on peut distinguer trois types de maintenance [Swa 76] qui correspondent à :

- la maintenance **corrective** qui consiste à corriger les fautes qui font que le système n'est pas conforme aux spécifications,
- la maintenance **adaptative** qui consiste à adapter le logiciel aux changements externes (de l'environnement) tels que des changements dans le système d'exploitation, du matériel, du compilateur...,
- la maintenance **perfective** dont le but est d'améliorer le logiciel ; elle inclut les changements internes souhaités par le concepteur ou par les utilisateurs qui demandent des modifications qui correspondent généralement à un changement de fonctionnalités.

En développement, le logiciel subit des corrections ainsi que des évolutions des spécifications.

I-1-1-1 *Corrections*

En développement, les corrections sont généralement introduites au fur et à mesure de leur identification ou par lots afin de minimiser le nombre de compilations et d'éditions des liens ce qui accélère le processus de correction. En cas de défaillance "bloquante", la correction est en principe introduite immédiatement.

En vie opérationnelle, lorsqu'une défaillance est survenue dans le logiciel, les opérations qui interviennent sont généralement :

- la reprise des traitements soit après relance du système avec une configuration des données d'entrée distincte de celle qui a conduit à la défaillance, soit après rechargement du logiciel si celui-ci a été affecté,
- la correction hors ligne de la (ou des) faute(s), sur une copie du logiciel que l'on exécute sur une autre machine, ou sur la même machine durant les périodes de non utilisation du logiciel.

Ce cas correspond à la majorité des logiciels et plus particulièrement à certaines applications aéronautiques (aussi bien certains logiciels embarqués que les moyens de contrôle au sol), aux télécommunications...

Ce type de correction sera appelé **correction différée** : la correction est effectuée après reprise des traitements.

Pour les systèmes dits critiques, ces opérations peuvent consister en l'arrêt des traitements, la reprise des traitements n'étant effectuée qu'après correction (sur le code

source ou sur exécutable) de la, ou des fautes. Ce type de correction sera appelé **correction immédiate** : les traitements sont suspendus pendant toute la durée des corrections.

Que la correction soit différée ou immédiate, une nouvelle **édition** du logiciel intégrant les corrections est introduite.

On peut remarquer que les mêmes types de politique de correction sont valables aussi bien en développement qu'en vie opérationnelle : correction immédiate ou différée, sur le source ou l'exécutable. Dans la suite, nous parlerons de corrections indépendamment de la phase du cycle de vie durant laquelle elles ont eu lieu.

La fréquence des corrections par rapport à celle des défaillances dépend à la fois de la nature de l'application, de l'organisation interne et des choix effectués par l'entreprise ainsi que de l'étape du cycle de vie.
Nous pouvons donner comme exemples les quatre situations suivantes :
- le logiciel est corrigé après chaque défaillance pour éviter d'activer plusieurs fois la même faute (redécouvertes) ; les corrections sont effectuées soit directement sur le code exécutable par "plâtrage" (patching) en attendant d'effectuer la modification sur le code source (ceci n'est possible que si le logiciel est en Assembleur), soit sur le source,
- les corrections sont effectuées uniquement après l'activation et l'identification de plusieurs fautes (correction par **lots**) ; dans ce cas elles sont introduites en principe au niveau du code source,
- la politique de correction dépend (au sein d'une même entreprise) de la nature et surtout des conséquences de la faute :
    - si les conséquences de la faute sont jugées graves, la correction (effectuée généralement sur le code exécutable) est immédiate,
    - si les conséquences de la faute sont jugées mineures, les corrections sont effectuées par lots ; la période peut être fixe et connue à l'avance ou fonction du nombre de fautes activées, ou conditionnée par celle de la maintenance adaptative ou perfective (on profite d'une évolution des spécifications pour introduire les corrections),
- afin de réduire le coût de la maintenance, les corrections ne sont introduites que par lots et en même temps que l'introduction de nouvelles spécifications.

En fait, même s'il existe une règle générale pour une entreprise donnée, elle n'est pas toujours appliquée de façon très rigoureuse car il y a souvent des contraintes qui font que la règle générale doit être adaptée au contexte spécifique d'un projet.

Il est à noter que même si la règle générale est d'introduire les corrections par lots en même temps qu'un changement de spécifications, il arrive souvent qu'après un changement de spécifications, si les conséquences de la défaillance sont graves, il est permis (et même conseillé) d'effectuer les corrections assez rapidement. Comme les spécifications évoluent constamment, les corrections en dehors de changement de spécifications ont lieu régulièrement, il en résulte dans la pratique une politique de correction continue.
Enfin il faut également signaler que la politique de correction évolue d'une phase à l'autre du développement.

Dans le cas où les corrections sont différées, on peut avoir plusieurs défaillances ayant la même origine : dues à la même faute avant qu'elle ne soit éliminée. Ces défaillances qu'on peut qualifier de redécouvertes [Ada 80] sont difficiles à prendre en compte, car souvent elles ne sont pas répertoriées, ou du moins les relevés de défaillances correspondants ne sont pas conservés. Leur prise en compte pourrait changer de façon significative les résultats de l'évaluation de la fiabilité du logiciel. Dans cette même référence, l'auteur donne la courbe donnant l'intensité de redécouverte par rapport à celle représentant l'intensité de défaillance sans tenir compte des redécouvertes : les deux courbes ont la même allure mais la courbe des redécouvertes est retardée dans le temps et son amplitude est un peu plus élevée. Le décalage dans le temps et l'augmentation de l'intensité sont fonction du taux d'occurrence des défaillances ; plus ce taux est élevé plus le décalage est réduit et plus l'intensité des redécouvertes est grande, il s'en suit que plus ce taux est grand plus l'influence est grande. Ceci montre que l'influence des redécouvertes est plus grande en développement qu'en vie opérationnelle stabilisée (puisque la fréquence des défaillances est plus élevée en développement), d'où la nécessité d'introduire les corrections assez rapidement en période de test afin de pas être gêné par ces redécouvertes.

I-1-1-2 *Maintenance adaptative et maintenance perfective*

Dans le cas de la maintenance adaptative, les fonctionnalités du logiciel sont inchangées, seules les interfaces du logiciel avec son environnement changent, il en résulte des changements des spécifications de l'environnement.
Dans le cas de la maintenance perfective, les fonctionnalités aussi bien que l'environnement peuvent changer : il peut y avoir suppression de fonctionnalités, introduction de nouvelles fonctionnalités ou substitution.
Que la maintenance soit adaptative ou perfective, les spécifications sont **changées**. Bien que la distinction entre ces deux types de maintenance soit primordiale pour la gestion de configuration de logiciel, leur influence au niveau des caractéristiques du logiciel est la même. Nous les regrouperons dans la suite sous le nom de **changement de spécifications**.

La fréquence de ce type de maintenance est uniquement fonction de l'évolution des besoins des utilisateurs et de contraintes externes telles que changement de normes, adaptation du produit à de nouveaux marchés..., et peut être variable durant la vie d'un logiciel.

La mise en œuvre des nouvelles spécifications donne lieu à une nouvelle **version** du logiciel.

Remarque :

Les techniques et les facilités mises en place pour aider et guider la correction et la maintenance du logiciel influencent considérablement le choix des politiques de correction et de maintenance ainsi que la durée de celles-ci. Il est donc essentiel de prévoir ce type de techniques dès la conception et la réalisation du logiciel afin de faciliter et d'accélérer les procédures de correction et de maintenance.

I-1-1-3 *Conclusions*

Nous avons introduit deux notions importantes au niveau de l'utilisateur d'un logiciel : la notion d'édition et celle de version. Une **édition** diffère de la précédente par les corrections introduites, une **version** diffère de la précédente par un changement de spécifications et éventuellement des corrections relatives à la version précédente. Il arrive cependant que l'utilisateur n'ait accès qu'aux versions, les éditions pouvant rester internes à l'entreprise.

Dans tout ce qui précède, les corrections peuvent être effectuées soit par l'équipe qui a conçu le logiciel soit par d'autres équipes ; comme nous nous intéressons plus particulièrement à l'influence des corrections sur les caractéristiques du logiciel nous ne prendrons pas en compte explicitement ce point bien qu'il soit important au niveau de l'organisation de l'entreprise.
Nous ne distinguerons pas non plus le cas où les corrections sont effectuées sur le source du cas où elles sont effectuées directement sur le code exécutable. Ceci est très important pour la gestion des configurations de logiciel (et plus généralement du système), mais n'a pas d'influence au niveau de notre étude.

I-1-2 Processus régissant le comportement d'un logiciel

Dans ce paragraphe, nous étudions de façon précise le processus de défaillance du logiciel et l'influence des différentes politiques de correction et de maintenance sur ses caractéristiques et, de ce fait, sur son comportement.

I-1-2-1 *Défaillances*

Conceptuellement, le comportement d'un logiciel peut être vu comme la transformation par un programme P d'un espace d'entrée E en un espace de sortie S (figure I-1).

Soient [Lap 83, Lap 88] :

- E_e le sous-espace des entrées erronées défini implicitement ou explicitement par les **spécifications,**
- E_{af} le sous-espace des entrées activant une faute.

E_e est étroitement lié aux spécifications, alors que E_{af} caractérise le programme lui même : si le programme P ne comportait pas de faute, le sous-espace E_{af} serait vide.

Une **défaillance survient** quand la trajectoire dans E intersecte le domaine de défaillance : sous-espace $E_f = E_e \cup E_{af}$. Ceci modélise le caractère aléatoire du comportement par rapport aux entrées, qui sera appelé dans la suite **incertitude sur les entrées** [Lit 80].

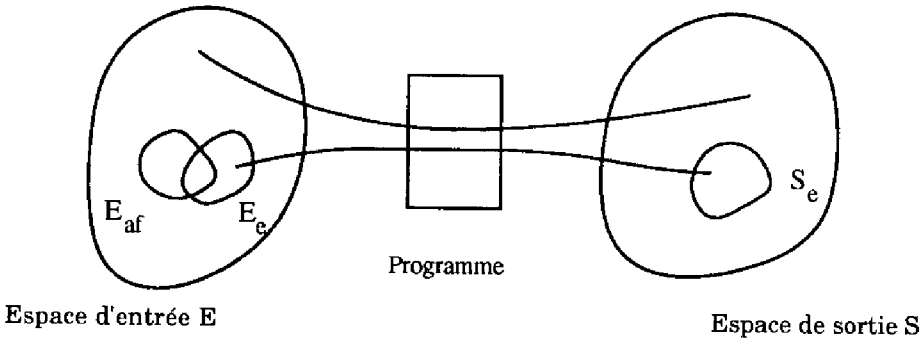


Figure I-1: Espaces des entrées erronées et des entrées activant une faute

I-1-2-2 *Corrections*

Soient deux éditions du même programme P(1) et P(2), écrites à partir des mêmes **spécifications**, et donc ayant le même espace des entrées E, le même espace de sortie S et le même sous-espace des entrées erronées E_e (figure I-2). Les deux éditions diffèrent par le sous-espace E_{af} dont l'union avec E_e donnera lieu à défaillance.

On voit donc apparaître une deuxième source d'incertitude, l'**incertitude sur le programme** [Lit 80] : la vie d'une version d'un logiciel peut être vue comme une séquence d'éditions P(1), P(2), ..., P(n), P(n+1),..., chacune d'elles différant de la précédente par les corrections qui lui ont été apportées. Le résultat des essais de correction étant lui-même imprévisible, il existe une incertitude sur la **relation** entre les comportements jusqu'à défaillance des éditions

successives : à la séquence d'éditions correspond une séquence $E_{af}(1)$, $E_{af}(2)$, ..., $E_{af}(n)$, $E_{af}(n+1)$, ... des sous-espaces d'entrées activant une faute. L'intention des programmeurs (et le souhait des utilisateurs) est d'éviter les régressions et donc d'obtenir $E_{af}(j-1) \supset E_{af}(j)$ (ou, ce qui est équivalent $E_f(j-1) \supset E_f(j)$) pour tout $j>1$, mais ils ne peuvent le garantir ; la séquence des programmes est donc elle-même aléatoire.

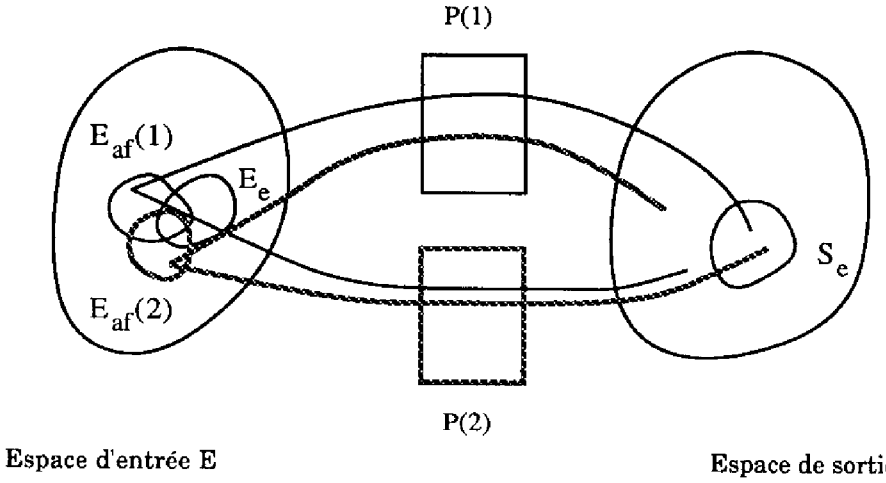


Figure I-2 : Espaces des entrées et sorties pour deux éditions P(1) et P(2) d'un programme.

I-1-2-3 *Changement de spécifications*

Suite au changement des spécifications, une nouvelle version du logiciel (correspondant à la mise en œuvre de ces nouvelles spécifications) est produite.

Le sous-espace E_e des entrées erronées défini par les spécifications peut ainsi être modifié. On voit ainsi apparaître une troisième source d'incertitude : **incertitude sur les spécifications**. La vie d'un logiciel peut être vue comme une séquence de versions issues de spécifications mises à jour (V_1, V_i ,...V_n) ; la version V_i (i = 1,2,..n) étant elle-même une séquence d'éditions matérialisées par les programmes ($P_{i.1}$, $P_{i.2}$, ... $P_{i.ni}$) issues des mêmes spécifications différant par les corrections effectuées (figure I-3).
La succession des versions forme ce qui sera appelé le **produit** logiciel.

I-1-2-4 *Profil d'utilisation*

Le profil d'utilisation constitue une source d'incertitude non négligeable. Il détermine en effet le choix des données fournies au programme parmi l'ensemble des données possibles E. Dans certains cas limites, en fonction de la charge, il peut modifier aussi bien le sous-espace E_e que E_{af}. Le profil d'utilisation détermine aussi la fréquence d'activation du logiciel et par là même la fréquence d'activation des fautes.

I-1-2-5 *Conclusions*

Cette approche de modélisation du mécanisme de défaillance du logiciel est intéressante car elle permet de bien distinguer :

- le comportement du logiciel jusqu'à la défaillance, où n'intervient que la première source d'incertitude, celle sur les entrées,
- le comportement d'une version, où n'interviennent que les deux premières sources d'incertitude, sur les entrées et sur les éditions successives du programme,
- le comportement durant le cycle de vie, où interviennent les trois sources d'incertitude, sur les entrées, sur les éditions et sur les versions successives du programme.

Dans le paragraphe suivant, nous nous intéresserons aux mesures de la sûreté de fonctionnement tenant compte de ces différentes sources d'incertitude.

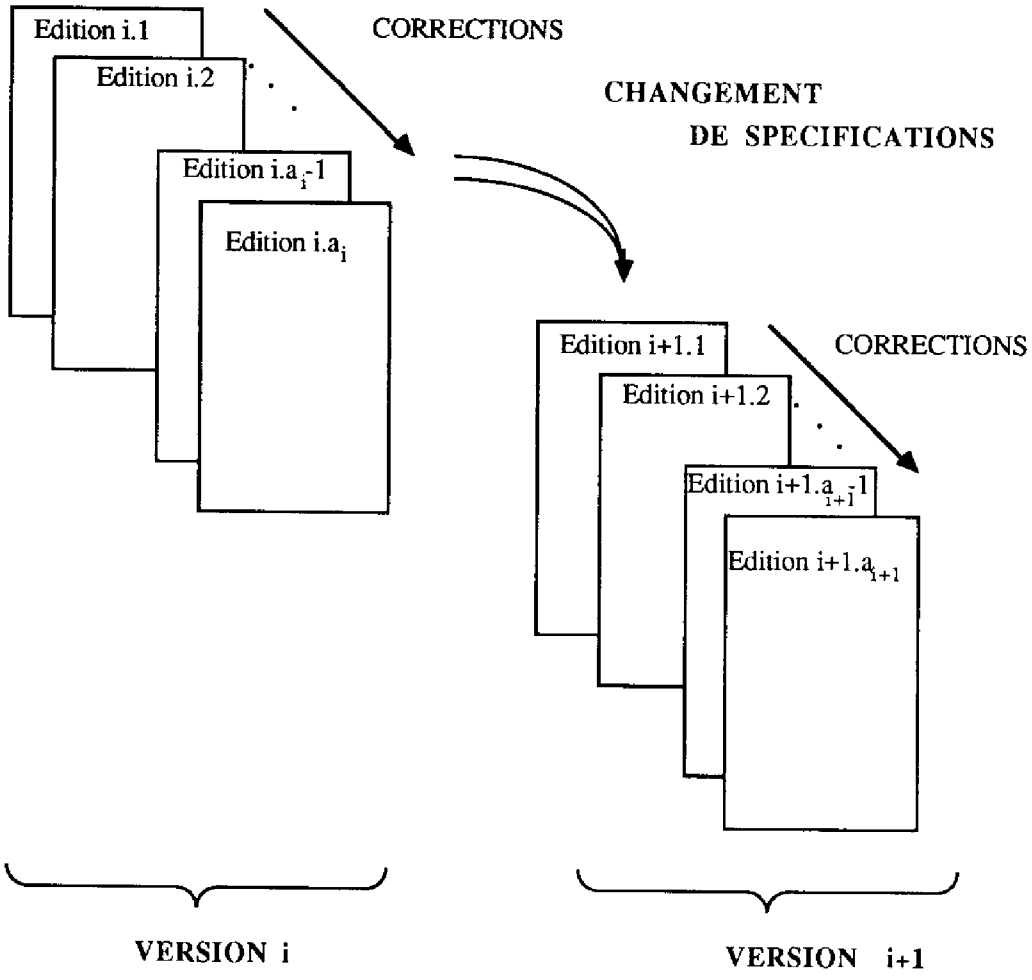


Figure I-3 : Vie d'un produit logiciel

I-2- DUREE DE VIE , EDITIONS ET VERSIONS POUR QUELQUES CAS REELS

Durant la vie d'un logiciel éditions et versions se succèdent à un rythme qui diffère d'un logiciel à un autre. L'ampleur et la fréquence des modifications sont étroitement liées à la nature du logiciel et à l'usage qui en est fait. Nous donnons ici, à titre indicatif, quelques chiffres concernant trois systèmes réels.

Le premier système correspond au système d'exploitation de l'IBM 360 [Leh 85]. Ce logiciel est utilisé sur des sites très dispersés avec des environnements d'utilisation très variés. Les équipes de développement et de maintenance sont très nombreuses et géographiquement distribuées. Le système a été observé sur une période de 11 ans et a subi des modifications fonctionnelles de façon continue afin de refléter les changements des environnements d'utilisation, les modifications des objectifs et les progrès technologiques. Le volume logiciel est passé de 1000 modules environ à 6000 modules environ vers la fin de la période considérée. La durée de vie de ce logiciel est de 20 ans au moins. Ce système a été observé dans le but d'étudier l'évolution du volume du programme (et donc des spécifications) en fonction du temps ; la croissance du nombre de modules est logarithmique : après une forte croissance au départ le nombre de modules a subi une croissance de plus en plus faible avant de se stabiliser vers la fin.

Le deuxième système correspond à l'autocommutateur téléphonique E-10 développé par Alcatel-CIT en utilisation en France et à l'étranger depuis à peu près sept ans. La durée de vie d'un tel logiciel se situe entre quinze et vingt ans. Il y a eu plus de sept changements de version au niveau de la France, soit en moyenne plus d'une version par an. Son adaptation à l'étranger a entraîné quinze changements de version.

Les principales modifications correspondent :
- au passage des concentrateurs analogiques aux concentrateurs numériques,
- à l'évolution des normes internationales des télécommunications,
- au changement de la numérotation téléphonique en France...

Les changements de spécification ont généralement affecté une seule fonction et ont entraîné en moyenne 5 à 10% d'augmentation du code.

Lors du développement de l'une des dernières versions, 90% des fautes corrigées dans le logiciel appartenaient au nouveau code, et 10% correspondaient aux corrections des défaillances signalées sur la version en cours d'utilisation. En vie opérationnelle, nous

n'avons pas remarqué une forte augmentation du nombre de défaillances suite à l'introduction des nouvelles versions.

Le troisième système correspond plus exactement à un ensemble de systèmes développés par l'Aérospatiale ; les données représentent des données moyennes et ont été enregistrées au cours du développement (ce dernier va de la phase des spécifications des besoins à la qualification système).
La taille moyenne du logiciel est de l'ordre de 500 Kilo-octets, la durée du développement est d'environ 8 ans et une cinquantaine de personnes ont participé au projet.

Trois types de modification ont eu lieu :
- correction des erreurs du logiciel,
- correction des erreurs de spécification,
- évolution dans les spécifications des besoins.

13% des modifications sont dues à des erreurs de logiciel, 21% aux erreurs de spécifications et 66% à des évolutions des spécifications.

Toute modification donne lieu à une nouvelle version et une version peut résulter du cumul des trois types de modifications (la notion d'édition n'existe pas).

I-3- MESURES DE LA SURETE DE FONCTIONNEMENT

Les mesures relatives au comportement d'un système jusqu'à défaillance sont largement étudiées dans la littérature [Bar 75, Cor 75, Lap 75,...] et s'appliquent au cas du logiciel ; elles seront uniquement énumérées dans ce paragraphe. Nous nous intéressons plus particulièrement aux mesures relatives au processus de défaillance durant le cycle de vie. Ces mesures seront définies dans un premier paragraphe ; le lien entre évolution de fiabilité et sens de variation de ces mesures sera effectué dans le deuxième paragraphe.

Quelles que soient les opérations effectuées (correction, maintenance adaptative ou perfective), nous nous trouvons dans une situation analogue à celle d'un processus de renouvellement "classique" (ou strict) : après introduction d'une nouvelle édition ou d'une version, le logiciel est "renouvelé". Si les caractéristiques du logiciel "renouvelé" sont identiques à celles du logiciel "remplacé" on parle d'un processus de **renouvellement strict** ; si le logiciel "renouvelé" n'est pas identique au précédent, on parle de processus de **renouvellement généralisé** [Kan 85, Lap 88].

La nature des mesures de la sûreté de fonctionnement ne dépend pas de la politique de maintenance considérée, mais leur évolution étant conditionnée par celle-ci, nous étudierons différentes situations. Pour les corrections, nous distinguerons les deux cas analysés au paragraphe I-1-1-1 : correction immédiate et correction différée.

I-3-1 Définition des mesures de la sûreté de fonctionnement

I-3-1-1 *Mesures relatives au comportement jusqu'à défaillance*

Le comportement du logiciel est caractérisé par l'étude de la variable aléatoire continue séparant la mise en œuvre du logiciel jusqu'à sa défaillance : soit T cette variable aléatoire.

T est caractérisée par un certain nombre d'attributs qui sont : la **fonction densité de probabilité** f(t), la **fonction de répartition** (ou la distribution) F(t), la **fonction de survie** (qui correspond à la **fiabilité** R(t)), la moyenne qui correspond au **temps moyen jusqu'à défaillance** noté MTTF, le **taux de défaillance** $\lambda(t)$ ...

Ces différents attributs sont liés et, mis à part le MTTF, il suffit d'en connaître un pour obtenir tous les autres ; ces liens sont résumés dans le tableau de la figure I-4.

	F(t)	R(t)	f(t)	$\lambda(t)$
F(t)	*	$1-R(t)$	$\int_0^t f(x)\,dx$	$1-\exp\int_0^t -\lambda(x)dx$
R(t)	1-F(t)	*	$\int_t^\infty f(x)\,dx$	$\exp\int_0^t -\lambda(x)dx$
f(t)	$\frac{dF(t)}{dt}$	$-\frac{dR(t)}{dt}$	*	$\lambda(t)\exp\int_0^t -\lambda(x)dx$
$\lambda(t)$	$\frac{1}{1-F(t)}\frac{dF(t)}{dt}$	$-\frac{1}{R(t)}\frac{dR(t)}{dt}$	$\frac{f(t)}{\int_t^\infty f(x)\,dx}$	*

Figure I-4 : Liens entre les mesures de sûreté de fonctionnement.

Notre objectif n'est pas de redéfinir tous ces attributs, nous précisons cependant la notion de taux de défaillance, appelé aussi taux de hasard, car cet attribut est souvent utilisé.

Taux de défaillance :

$\lambda(t)\, dt$ = Probabilité { une défaillance survient entre t et t+dt | le logiciel est non défaillant sur [0, t] } (I-1)

I-3-1-2 *Mesures relatives au comportement après reprise d'exécution*

On suppose que les T_i constituent une suite de variables aléatoires indépendantes (processus de renouvellement), les mesures permettant de caractériser le processus de défaillance d'un logiciel en tenant compte des reprises des traitements après défaillance sont :

- **le nombre cumulé de défaillances**, N(t), survenues sur l'intervalle [0,t],
- **le nombre moyen de défaillances** sur l'intervalle [0,t] appelé aussi **moyenne** du processus ou **fonction de renouvellement**,

$$M(t) = E[N(t)], \tag{I-2}$$

- **le taux d'occurrence des défaillances** (ou **densité de renouvellement**) correspond au nombre moyen de défaillances sur un intervalle de temps dt proche de t,

$$\text{ROCOF} = \lim_{dt \to 0} \frac{M(t+dt) - M(t)}{dt} = M'(t) \tag{I-3}$$

- l'**intensité de défaillance** ,

$$h(t) = \lim_{dt \to 0} \frac{\Pr\{N(t+dt) - N(t) \geq 1\}}{dt} \tag{I-4}$$

Un processus stochastique est dit ordonné ou régulier si la probabilité d'occurrence simultanée de deux événements est négligeable, ce qui est le cas du processus de défaillance d'un logiciel.

Dans ce cas on a :

$$\Pr\{N(t+dt) - N(t) \geq 2\} = o(dt) \tag{I-5}$$

et l'intensité de défaillance devient, pour $dt \to 0$:

$h(t)\, dt$ = { Probabilité qu'une défaillance survienne entre t et t+dt }. (I-4')

Pour toutes les mesures précédentes, la variable t mesure le temps de fonctionnement du logiciel : les durées d'arrêt des traitements suite à une défaillance sont soit supposées négligeables soit non prises en compte. Si les corrections sont effectuées hors ligne après relance du logiciel et que le logiciel est utilisé de façon continue, le temps de fonctionnement correspond au temps calendaire (temps courant).

En vie opérationnelle, si les corrections sont effectuées avant relance ou si la durée de réinitialisation du système après défaillance du logiciel n'est pas négligeable, la sûreté de fonctionnement du logiciel est plus ou moins affectée par la durée de ces arrêts ; dans ce cas, la **disponibilité** constitue une mesure intéressante. En effet l'indisponibilité du système entraînée par les arrêts suite à une défaillance du logiciel peut ne pas être négligeable par rapport à l'indisponibilité due aux défaillances du matériel.

Commentaires :

1) Dans le cas général [ROS 70], il a été montré que, pour un processus stochastique régulier, le ROCOF est égal à l'intensité de défaillance. Dans ce qui suit h(t) sera aussi bien appelé intensité de défaillance que taux d'occurrence des défaillances selon le contexte :

$$\text{ROCOF} = M'(t) = h(t)$$

$$\text{Moyenne} = M(t) = \int_0^t h(x)\,dx = H(t)$$

2) D'un point de vue formel, la différence entre intensité de défaillance h(t) et taux de défaillance $\lambda(t)$ réside dans le fait que ce dernier correspond à une probabilité de défaillance conditionnelle. D'un point de vue pratique, le **taux de défaillance** caractérise **une** défaillance, alors que l'**intensité de défaillance** caractérise le **processus** de défaillance (flux des défaillances) indépendamment du nombre de défaillances qui ont eu lieu entre l'instant initial et l'instant t. L'intensité de défaillance intègre le processus de modification du logiciel.

Pour un processus poissonnien homogène, toutes ces notions sont confondues ; en effet dans ce cas les éditions sont supposées "identiques", ou du moins ayant le même taux de défaillance constant. Ce taux est aussi égal au taux d'occurrence des défaillances successives (ROCOF) qui est lui même égal à l'intensité de défaillance, c'est ce qui explique les confusions qui sont faites généralement entre taux et intensité de défaillance.

Correction immédiate

Le taux d'occurrence des défaillances (ROCOF) caractérise la perception par un observateur aussi bien du flux des défaillances que celui des corrections puisque, dans ce cas, ces deux processus sont liés : à une défaillance correspond une correction. Le nombre de défaillances N(t) correspond aussi au nombre d'éditions du programme.

Correction différée

Dans ce cas le logiciel est caractérisé à la fois par la fréquence des défaillances et celle des corrections. En fait, les traitements sont repris soit après occurrence d'une défaillance et relance (avec ou sans chargement du logiciel à nouveau), soit après introduction d'une nouvelle édition. N(t) est alors défini comme étant le nombre de reprises des traitements (avec ou sans introduction d'une nouvelle édition), les mesures précédemment définies restent valables (le renouvellement doit être pris dans le sens reprise des traitements).
Pour la disponibilité, il faut tenir compte à la fois des durées d'arrêt avant relance et des durées d'arrêt pour le chargement d'une nouvelle édition.

Changement de spécifications

Dans ce cas le logiciel est caractérisé par la fréquence des défaillances, celle des corrections et celle d'introductions de nouvelles versions. Néanmoins les mesures sont les mêmes que pour un logiciel à spécifications inchangées. Le changement de spécifications entraîne la modification simultanée des sous-espaces E_e et E_{af}, ce qui entraîne une variation du ROCOF (ou intensité de défaillance) qui peut se traduire par une **discontinuité** ; cette variation est plus ou moins importante selon la nature des changements.

Relation entre taux de défaillance et intensité de défaillance

La relation entre le taux de défaillance $\lambda(t)$ et l'intensité de défaillance $h(t)$ est fonction de l'environnement d'utilisation et de la politique de maintenance adoptée ; elle est généralement très complexe et ne peut être établie de façon analytique que dans des cas particuliers. Deux ensembles de cas seront considérés dans le chapitre II :
- les corrections sont effectuées après occurrence d'un certain nombre de défaillances,
- le processus de défaillance est modélisé par un processus de Poisson non homogène.

I-3-2 Evolution des mesures de la sûreté de fonctionnement

D'après ce qui précède, on voit que durant le cycle de vie, le logiciel peut être caractérisé par l'intensité de défaillance h(t). Comme l'intensité de défaillance est aussi égale au taux d'occurrence des défaillances (ROCOF), un estimateur statistique de h(t) consiste à prendre le nombre de défaillances ayant eu lieu sur un intervalle de temps court au voisinage de t. Dans la pratique, le temps total d'observation du logiciel est divisé en intervalles de même longueur et la suite des nombres de défaillances sur ces intervalles constitue un estimateur de h(t).
Une variation de la fiabilité du logiciel se traduit par une variation du nombre moyen de défaillances sur ces intervalles de temps, qui se traduit par une variation de l'intensité de défaillance.

Une variation de fiabilité peut résulter :

- d'un changement du **domaine de défaillance** du logiciel : variation du sous-espace E_e des entrées erronées défini par les spécifications suite à des changements des spécifications de l'environnement et/ou des fonctionnalités, ou du sous-espace E_{af} des entrées activant une faute suite à des corrections ou des changements de spécifications,
- de la variation du **profil d'utilisation** :
  - en vie opérationnelle, le profil d'utilisation est fonction des besoins spécifiques de chaque utilisateur et de la charge du système,
  - en développement, le profil d'utilisation est étroitement lié aux différentes phases par lesquelles passe le logiciel et à l'intérieur d'une même phase, il est conditionné par les différents tests que subit le logiciel et le nombre de personnes chargées du test (qui peut varier considérablement)...

Le nombre de copies disponibles pour le test et le nombre de sites où est implanté ce logiciel en vie opérationnelle peut influencer considérablement la vitesse d'évolution de la fiabilité, puisqu'un grand nombre de copies permet d'exercer un plus grand nombre de configurations dans le domaine d'entrée.

L'influence de la nature du test appliqué au logiciel est schématisée par la figure I-5. Les intervalles de temps entre défaillances sont très petits au début de l'application d'un test et sont de plus en plus grands au fur et à mesure qu'on élimine les fautes activées par ce test. L'intervalle entre défaillances diminue brusquement quand on passe au test suivant.

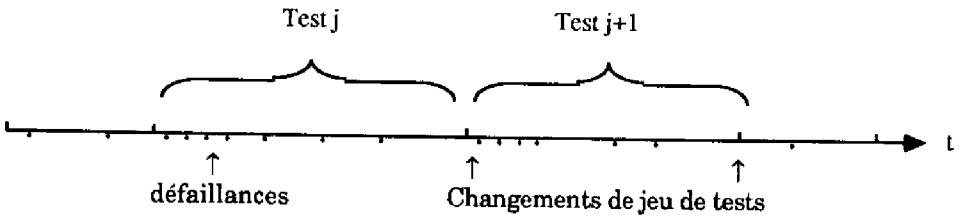


Figure I-5 : Evolution des intervalles entre défaillances en fonction des tests appliqués.

Comme exemple on peut citer les données de défaillances prélevées sur un système de contrôle de combustibles nucléaires irradiés [Fav 85]. Ces données confirment bien le fait que des diminutions brusques des intervalles entre défaillances sont observées à chaque changement de jeu de tests, au moment d'un nouveau programme et lors de la modification de l'environnement.

Nous étudierons dans ce qui suit le sens de variation de la fiabilité et de l'intensité de défaillance pour une version donnée d'un logiciel (spécifications inchangées) et pour une suite de versions (changement de spécifications).

I-3-2-1 *Variation de la fiabilité à spécifications inchangées*

Pour un profil d'utilisation donné, le sous-espace E_e est le même d'une édition à l'autre, seul le sous-espace E_{af} varie. On a vu qu'après la (j-1) ième correction (édition) on espère avoir $E_{af}(j-1) \supset E_{af}(j)$; ceci n'est vrai que s'il n'y a pas introduction de nouvelles fautes dans le logiciel. Au niveau de l'utilisateur, ceci se traduit par :

- des intervalles entre défaillances (T_i) stochastiquement croissants ou —ce qui est équivalent— des taux de défaillance stochastiquement décroissants ; la croissance stochastique d'une suite de variables aléatoires Ti dont la fonction de répartition est notée $F_{Ti}(t)$, est définie par :

  $T_i \leq_{st} T_j \Leftrightarrow F_{Ti}(t) \leq F_{Tj}(t)$ pour tout $i \geq 1$, $j>i$ et $t>0$, (I-6)
- un nombre moyen de défaillances par unité de temps décroissant, ce qui signifie une intensité de défaillance décroissante.

On parle alors de **croissance de fiabilité**.

Une **décroissance de fiabilité** peut avoir lieu suite à :

- des corrections erronées qui font que la relation $E_{af}(j-1) \supset E_{af}(j)$ n'est plus vérifiée,
- des changements dans le profil d'utilisation qui entraînent des intervalles entre défaillances plus petits, bien qu'au niveau du logiciel on ait bien $E_{af}(j-1) \supset E_{af}(j)$ et que le nombre de fautes résiduelles diminue.

Elle se traduit par des taux de défaillance stochastiquement croissants et une intensité de défaillance croissante.

En fait, la combinaison de ces deux processus (corrections erronées et changement du profil d'utilisation) peut entraîner une alternance de périodes de croissance et de décroissance de fiabilité. Un exemple bien connu de cette alternance est matérialisé par la courbe de la figure I-6 qui donne le nombre de défaillances observées par tranches de deux mois d'un logiciel de Spacelab pour une période d'intégration et de vérification du logiciel de deux ans et demi [Rob 83] ; un total de 2200 fautes ont été corrigées.

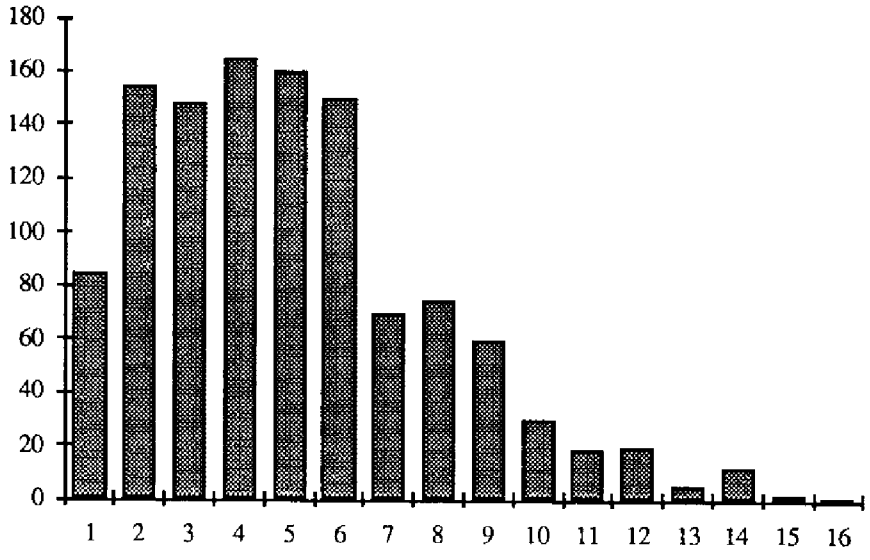


Figure I-6 : Evolution du nombre de défaillances par période de 2 mois pour un logiciel de Spacelab.

Cette **courbe en forme de cloche** est usuelle pour des logiciels en développement [Rem 82, Sab 87] et donne l'allure de l'intensité de défaillance : après une période de décroissance de fiabilité où le nombre de défaillances par unité de temps croît, le logiciel atteint une période de croissance globale de la fiabilité.

La décroissance résulte de la combinaison de plusieurs phénomènes :
- un logiciel qui n'a pas été utilisé ne peut avoir révélé de fautes, au moment de son utilisation le nombre de fautes activées ne peut qu'augmenter,
- l'effort de test est variable durant le développement [Par 80],
- la dépendance (ou le lien) entre certains types de fautes qui se traduit par le masquage de certaines fautes par d'autres : tant que ces dernières ne sont pas éliminées les fautes

masquées ne peuvent pas être activées et dès que les fautes masquantes sont corrigées les fautes masquées deviennent détectables [Ohb 84], [Sol 88a],
- les délais entre les défaillances et les corrections correspondantes varient au cours du développement et un grand décalage au départ peut constituer un handicap surtout en présence de fautes liées.

Enfin signalons que même à l'intérieur de la période de croissance de fiabilité, le logiciel peut passer par des périodes de décroissance locale de la fiabilité ayant pour origine des mauvaises corrections ou des perturbations dans l'environnement d'utilisation ou à de nouvelles utilisations entraînant des conditions d'entrée très particulières et donc rares.

I-3-2-2 *Variation de la fiabilité et changement de spécifications*

L'expérience montre que l'amélioration de la fiabilité ne peut être garantie après changement de version.
A ce niveau, il faut distinguer deux cas :

- en développement, les parties correspondant aux modifications introduites (substitution ou adjonction de fonctionnalités) ne sont pas aussi fiables que les précédentes, ce qui peut entraîner une décroissance de fiabilité, et donc une augmentation de l'intensité de défaillance se traduisant par une discontinuité ; si E_e n'est modifié que légèrement on peut espérer que la discontinuité de l'intensité de défaillance due au changement de version n'est pas très forte,
- en vie opérationnelle, la décroissance de fiabilité peut être réduite, en effet une grande partie des fautes dans les nouvelles parties est éliminée au cours du développement de ces parties ; on pourrait même avoir l'effet inverse dans le cas où les corrections des fautes activées dans la version précédente sont introduites par lots en même temps qu'un changement de version : l'augmentation de h(t) due au changement de version peut être compensée par l'amélioration due aux corrections des parties qui existaient déjà dans le programme.

De façon générale, en vie opérationnelle, en supposant que l'évolution du produit logiciel s'est effectuée progressivement, en tenant compte au niveau de chacune de ses versions des résultats de l'étude de fiabilité effectuée sur la version précédente, on peut espérer qu'il y ait amélioration globale du produit logiciel au cours du temps et que la courbe correspondant au lissage de l'intensité de défaillance soit globalement décroissante. Ceci est illustré par la figure I-7 où les courbes en continu représentent les intensités de défaillances des versions successives et la courbe en pointillé donne l'intensité de défaillances du produit.

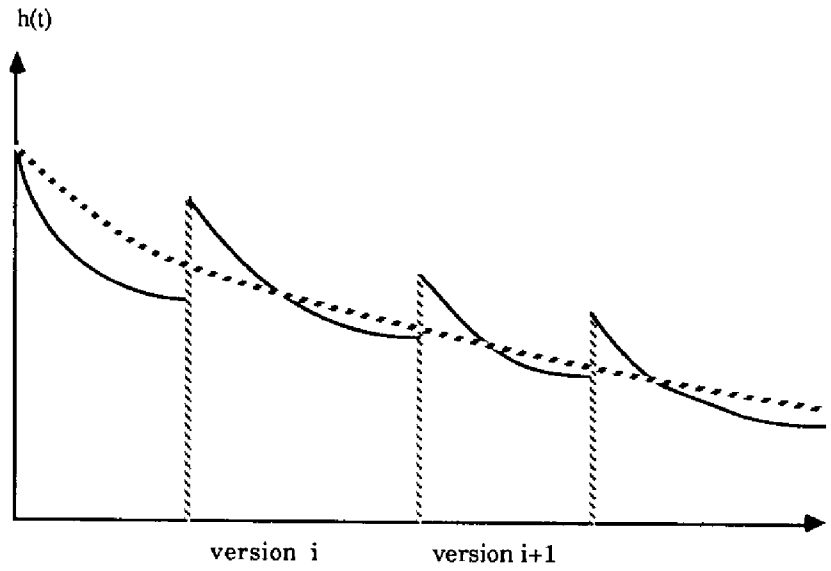


Figure I-7 : Intensité de défaillance d'un produit logiciel et de ses versions.

Si l'on s'intéresse à la fiabilité du logiciel à un instant donné il faut tenir compte de cette discontinuité ; dans la pratique ceci pourra être réalisé en ne prenant en compte que les données relatives à la version considérée.

Si en revanche, l'on s'intéresse à l'évolution de la fiabilité du produit on peut approcher les courbes intensité de défaillance des différentes versions par la courbe en pointillé sur la figure I-7 qui donne l'intensité de défaillance du produit. Ceci est d'autant plus vrai qu'en pratique les spécifications évoluent progressivement, ce qui entraîne une expansion régulière des logiciels correspondants [Leh 85]. Il est à noter que l'étude effectuée dans cette dernière référence sur trois logiciels montre que le taux d'expansion d'un logiciel décroît régulièrement tout au long de son cycle de vie. On peut donc espérer que son influence devient de moins en moins importante en vie opérationnelle

I-3-3 Conclusions

Dans ce paragraphe nous avons défini les différentes mesures de la sûreté de fonctionnement. Nous avons distingué deux types de mesures : celles relatives à l'occurrence d'une défaillance (taux de défaillance, fiabilité...) et celles relatives au cycle de vie : intensité (ou taux d'occurrence des défaillances), fonction de renouvellement, disponibilité. Ces mesures permettront d'évaluer la sûreté de fonctionnement du logiciel à l'aide des modèles qui seront établis dans le chapitre suivant. Nous avons aussi établi le lien entre le sens de variation de ces mesures et l'évolution de la fiabilité du logiciel : une croissance de "fiabilité" est caractérisée par des taux de défaillance décroissants et donc une intensité de défaillance décroissante.

L'analyse de l'évolution de l'intensité de défaillance pour une version du logiciel (à spécifications constantes) et pour des versions successives du logiciel (changement de spécifications) permet d'avoir des éléments concrets pour établir les modèles de comportement du logiciel.

I-4- CONCLUSIONS

Dans ce chapitre nous avons :

- identifié les facteurs qui caractérisent un logiciel et analysé leur influence sur son comportement,
- analysé quelques informations relatives à des systèmes réels concernant la durée de vie des éditions, des versions et l'ampleur des modifications subies par le logiciel au cours de son évolution,
- défini les mesures de sûreté de fonctionnement d'un logiciel et étudié leurs évolutions dans le temps.

Nous avons introduit deux notions fondamentales qui interviennent durant la vie d'un logiciel : les notions d'édition et de version. Durant sa vie, un produit logiciel passe par un certains nombre de versions ; le passage d'une version à la suivante est entraîné par le changement des spécifications. Une version est constituée d'une suite d'éditions, chacune différant de la précédente par les corrections effectuées.

L'étude de la sûreté de fonctionnement peut être effectuée soit pour toute ou partie d'une version (spécifications inchangées) soit pour un ensemble de versions soit pour l'ensemble des versions (le produit).

L'évolution de la fiabilité et de l'intensité de défaillance d'un logiciel a été étudiée pour une version donnée et pour le produit.

L'analyse qualitative des sens de variation respectifs de la fiabilité et de l'intensité de défaillance effectuée dans ce chapitre sera complétée par une analyse quantitative au chapitre II.

CHAPITRE II

MODELISATION

DE LA CROISSANCE DE LA SURETE DE FONCTIONNEMENT

INTRODUCTION

Le but de ce chapitre est de modéliser le comportement d'un logiciel durant le cycle de vie en tenant compte de diverses politiques de correction et de maintenance. Des **modèles de connaissance** caractérisant le comportement d'un logiciel seront établis. L'expression des mesures de la sûreté de fonctionnement relatives à ces modèles étant très complexe, ces modèles seront approchés par des **modèles d'action** (appelés modèles de croissance de fiabilité). Les expressions des mesures de la sûreté de fonctionnement relatives aux modèles d'action sont plus simples.

L'avantage des modèles de connaissance réside dans le fait qu'ils permettent de mettre en évidence les différents phénomènes mis en jeu afin de distinguer leur influence séparément (sans aucun souci quant à la complexité des modèles qui en résultent). Ils permettent entre autres d'analyser l'influence sur l'évolution de la fiabilité d'un logiciel :

- du profil d'utilisation,
- de politiques de correction différentes,
- du grand nombre de copies ou d'instances du même logiciel en utilisation sur différents sites.

Pour établir les modèles de connaissance nous considérerons progressivement différentes hypothèses de correction du logiciel, ces hypothèses correspondent à des politiques de correction très régulières et bien définies à l'avance. Les résultats issus de ces modèles servent à montrer l'impact de ces hypothèses sur la variation de la fiabilité du logiciel.

L'avantage des modèles d'action réside dans le fait qu'ils sont en général des modèles compacts plus facilement maniables tout en permettant de rendre compte des mêmes phénomènes de façon implicite. Ils permettent d'évaluer les mesures de la sûreté de fonctionnement d'un logiciel en évolution à l'aide d'expressions moins complexes que celles obtenues à partir des modèles de connaissance. Ce sont des modèles qui, mathématiquement, sont plus facilement exploitables. Quelques modèles de croissance de fiabilité existants seront présentés en mettant l'accent sur la nature de la variation des mesures de la sûreté de

fonctionnement qu'ils permettent de modéliser beaucoup plus que sur leurs hypothèses de base qui sont certes importantes mais qui n'entrent pas dans les objectifs de ce chapitre. Les aspects de précision et de validité des modèles d'action seront explicités dans le chapitre IV.

Ce chapitre est divisé en deux paragraphes :
- établissement des modèles de connaissance relatifs aux différentes politiques de correction et de maintenance présentées au chapitre précédent,
- présentation de quelques modèles d'action et plus particulièrement du modèle hyperexponentiel développé au LAAS.

II-1- MODELES DE CONNAISSANCE

L'objectif est de modéliser le comportement d'un logiciel durant le développement et la vie opérationnelle : nous considérerons d'abord de façon approfondie les modèles couramment appelés "modèles de croissance de fiabilité", le cas de la disponibilité sera ensuite considéré. L'appellation modèle de fiabilité est due au fait que les temps d'arrêt suite à une défaillance ne sont pas pris en compte par ces modèles : la correction n'intervient pas de façon explicite par sa durée mais par l'intermédiaire de ses effets au niveau du logiciel.

II-1-1 Modèles de croissance de fiabilité

La "fiabilité" sera étudiée pour différentes politiques de correction et de maintenance : nous commencerons par étudier le comportement d'un logiciel entre deux défaillances, nous modéliserons ensuite un logiciel avec correction immédiate et un logiciel avec correction différée ; enfin nous étudierons le cas de la maintenance adaptative et perfective. L'évolution de la fiabilité étant perçue au niveau de l'intensité de défaillance, c'est cette dernière mesure qui sera évaluée pour les différents cas considérés.

II-1-1-1 *Comportement entre défaillances*

Le logiciel est caractérisé par la variable aléatoire intervalle entre défaillances, la première question qui se pose alors est la nature de sa distribution.
L'hypothèse la plus couramment admise est celle d'un taux de défaillance constant [Jel 72, Tri 75, Lap 83, Lap 84a,b] ; une discussion détaillée sur la validité de cette hypothèse se trouve dans les deux dernières références. Ce taux ne peut être considéré constant que pour un logiciel opérant sous un profil d'utilisation et dans des conditions d'environnement et de charge stables. Une variation de l'un quelconque de ces facteurs affecte le taux de défaillance qui peut alors varier en fonction du temps.

Si λ_j est le taux de défaillance du logiciel relatif à la j ième défaillance, la densité de probabilité est alors donnée par :

$$f_j(t) = \lambda_j \exp(-\lambda_j t) \qquad \text{(II-1)}$$

II-1-1-2 *Correction immédiate*

Avec l'hypothèse d'un taux de défaillance constant, le processus de défaillance peut être modélisé par un processus de Poisson par morceaux, aucune relation n'étant supposée entre les taux successifs. On suppose par contre qu'au bout d'un certain nombre d'éditions (corrections), le taux de défaillance atteint une valeur limite non nulle [Lap 84a]. Cette hypothèse signifie soit qu'il n'y a plus de correction effectuée sur le logiciel avant introduction d'une nouvelle version, soit que les corrections encore effectuées n'affectent pas le taux de défaillance de façon significative. Cette hypothèse est confirmée expérimentalement pour des logiciels en opération [Ada 80, Sab 87].
Soit r le nombre d'éditions au bout duquel le taux de défaillance est invariant, et λ_r le taux correspondant ; λ_r est appelé **taux de défaillance résiduel**. Ce taux correspond au taux de défaillance du logiciel en régime stabilisé.
La durée de la période d'évolution — et de ce fait la valeur de r — dépend à la fois du logiciel, de l'environnement d'utilisation et du niveau de fiabilité atteint par rapport à celui visé pour la vie opérationnelle (par les spécifications).

Notre but est d'établir l'expression de l'intensité de défaillance en fonction des taux de défaillance successifs. Pour cela nous introduisons les notations suivantes :

- T_i, intervalle de temps entre la (i-1) ième et la i ième défaillance,
- τ_n, instant d'occurrence de la n ième défaillance : $\tau_n = \sum_{i=1}^{n} T_i$,
- $\phi_n(t)$, fonction densité de probabilité de τ_n,
- $\Phi_n(t)$, fonction de répartition de τ_n.

En généralisant la théorie du renouvellement au cas où les variables aléatoires ne sont pas identiquement distribuées [Kan 85, Lap 88], et en suivant un raisonnement similaire à celui utilisé pour l'établissement de la fonction de renouvellement pour un processus de renouvellement classique [Cox 66, Gne 72], on obtient :

$$\phi_n(t) = f_1(t) * f_2(t) * \ldots f_j(t) * \ldots * f_n(t) \qquad \text{pour } n \le r \qquad \text{(II-2)}$$

$$\phi_n(t) = f_1(t) * f_2(t) * \ldots f_j(t) * \ldots * f_r(t)^{*[n-r+1]} \qquad \text{pour } n > r \qquad \text{(II-3)}$$

où le symbole * dénote l'opération de convolution (la convolution de deux fonctions f(t) et g(t) étant définie par : $f(t) * g(t) = \int_0^t f(x)\, g(t-x)\, dx$), et $f_r(t)^{*[n-r+1]}$ la [n-r+1] ième convolution de $f_r(t)$ par elle même.

Par définition, on a :

$$P\{N(t) > n\} = P\{ \tau_n < t \} = \Phi_n(t) \qquad \text{(II-4)}$$

d'où : $P\{N(t) = n\} = P\{ \tau_n < t < \tau_{n+1} \} = \Phi_n(t) - \Phi_{n+1}(t)$

La fonction de renouvellement H(t) est donnée par :

$$H(t) = \sum_{n=1}^{\infty} n\, P\{N(t) = n\} = \sum_{n=1}^{\infty} n\, [\, \Phi_n(t) - \Phi_{n+1}(t)\,]$$

$$H(t) = \sum_{n=1}^{\infty} n\, \Phi_n(t) - \sum_{n=1}^{\infty} (n-1)\, \Phi_n(t) = \sum_{n=1}^{\infty} \Phi_n(t) \qquad \text{(II-5)}$$

L'intensité de défaillance h(t) est égale à la densité de renouvellement :

$$h(t) = \frac{d\, H(t)}{dt} = \sum_{n=1}^{\infty} \phi_n(t) \qquad \text{(II-6)}$$

En remplaçant $\phi_n(t)$ pour $n \le r$ par (II-2) et pour $n > r$ par (II-3) et après développement de (II-6), on a :

Pour r = 2

$$h(t) = \lambda_2 + (\lambda_1 - \lambda_2) \exp -(\lambda_1 t) \qquad \text{(II-7)}$$

Pour r > 2

$$h(t) = \lambda_r + \sum_{i=1}^{r-1} \alpha_i \exp(-\lambda_i t) \qquad \text{(II-8)}$$

avec :

$$\alpha_i = \left(\sum_{k=i}^{r-2} \lambda_i \prod_{j=1;\ j \neq i}^{k} \frac{\lambda_j}{\lambda_j - \lambda_i} \right) + (\lambda_i - \lambda_r) \prod_{j=1;\ j \neq i}^{k} \frac{\lambda_j}{\lambda_j - \lambda_i} \qquad \text{(II-9)}$$

et $\prod_{j=a}^{b} x_j = 1$ pour $b < a$

Quand les λ_i sont décroissants de λ_1 à λ_r, h(t) est une fonction continue, décroissante de λ_1 à λ_r.
En supposant une croissance de fiabilité monotone, l'allure de la courbe des taux de défaillance est donnée par la figure II-1-a, celle de h(t) par la figure II-1-b.

On peut aussi vérifier que si les λ_i sont croissants puis décroissants, l'intensité de défaillance a bien la forme d'une cloche (figure II-2).

II-1-1-3 *Correction différée*

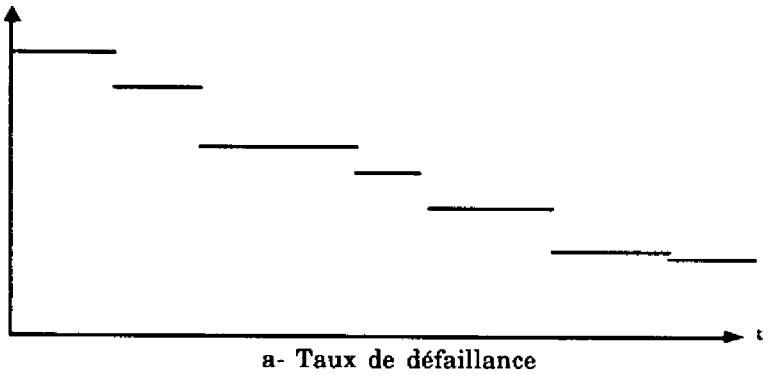
Pour des raisons d'efficacité et de réduction du coût de la maintenance, après défaillance d'un logiciel, la reprise des traitements est effectuée par relance du logiciel et les corrections sont effectuées hors ligne sur une copie. Une fois ces corrections terminées, il y a introduction d'une nouvelle édition.

Plusieurs défaillances peuvent ainsi avoir lieu pour une même édition avant que les corrections correspondantes ne soient effectuées.

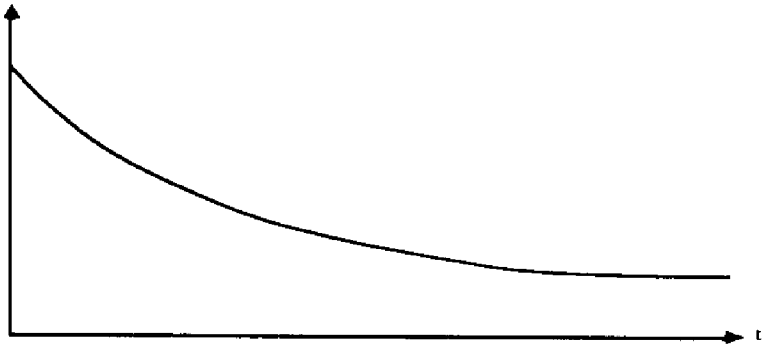
Après relance, le logiciel n'ayant pas été modifié mais juste relancé avec des données d'entrée différentes de celles qui ont entraîné la défaillance (ou rechargé), le sous-espace E_{af} des entrées activant une faute et E_e le sous-espace des entrées erronées défini par les spécifications sont inchangés. On suppose donc que le taux de défaillance est inchangé après relance.
λ_j est alors défini comme le taux de défaillance de l'édition j et $f_j(t)$ la densité de probabilité associée.

Soient :

- $T_{j,i}$, l'intervalle de temps entre la (i-1) ième et la i ième défaillance de l'édition j,
- Y_j, la durée de la j ième édition,
- S_j, l'instant d'introduction de la (j+1) ième édition,
- a_j, le nombre de défaillances de l'édition j,
- τ_n, l'instant d'occurrence de la n ième défaillance,
- k, le nombre de changements d'édition avant la n ième défaillance,
- a, le nombre de défaillances depuis l'introduction de l'édition (k+1) jusqu'à l'instant τ_n,
- $\phi_n(t)$, la fonction densité de probabilité de τ_n.

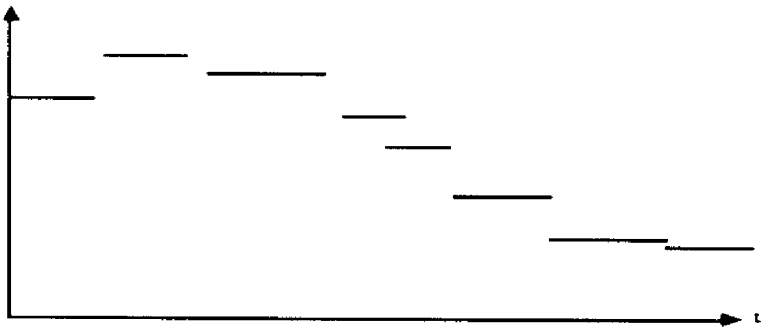


a- Taux de défaillance

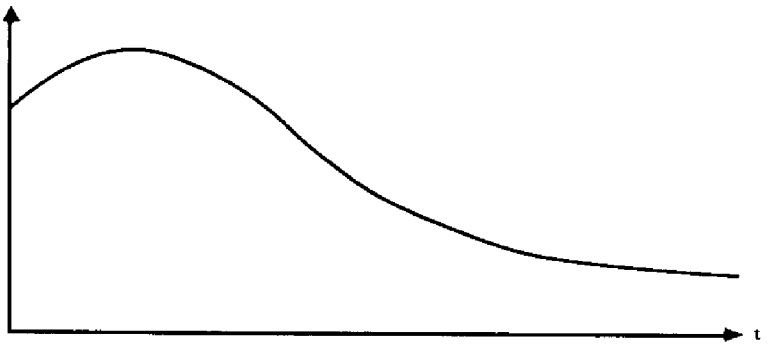


b- Intensité de défaillance

Figure II-1 : Taux $\lambda(t)$ et intensité de défaillance $h(t)$ à croissance de fiabilité monotone



a- Taux de défaillance



b- Intensité de défaillance

Figure II-2 : Taux $\lambda(t)$ et intensité de défaillance $h(t)$ à croissance de fiabilité variable

Deux situations sont possibles : soit on met à profit l'occurrence d'une défaillance pour introduire la nouvelle édition (intégrant les corrections relatives aux défaillances antérieures) avant la reprise des traitements, soit l'édition est introduite dès qu'elle est prête indépendamment des instants d'occurrence des défaillances. Ces deux cas seront successivement considérés dans ce qui suit.

II-1-1-3-1 Cas où l'introduction d'une édition est effectuée après défaillance

En supposant que l'introduction d'une nouvelle édition a lieu juste après l'occurrence d'une défaillance, la relation entre les différents intervalles de temps est représentée sur la figure II-3.

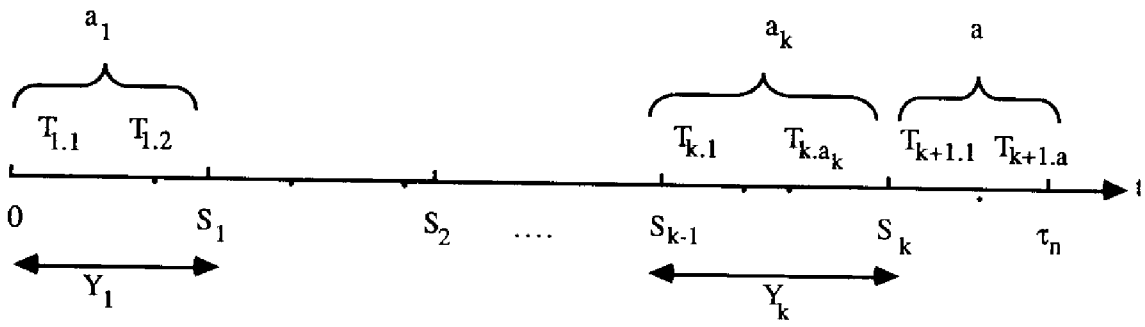


Figure II-3 : Relation entre les différents intervalles de temps.

La signification des différents intervalles de temps est la suivante :

$$Y_j = \sum_{i=1}^{a_j} T_{j,i} \,, \qquad \text{(II-10)}$$

$$S_j = \sum_{i=1}^{j} Y_i \,, \qquad \text{(II-11)}$$

$$\tau_n = S_k + \sum_{i=1}^{a} T_{k+1,i} \,, \qquad \text{(II-12)}$$

$$n = \sum_{j=1}^{k} a_j + a \qquad \text{(II-13)}$$

En tenant compte de ces relations, les expressions (II-2) et (II-3) deviennent :

$$\phi_n(t) = f_1(t)^{*[a1]} * f_2(t)^{*[a2]} * \dots f_j(t)^{*[aj]} * \dots * f_{k+1}(t)^{*[a]}, \text{ pour } k+1 \le r \quad \text{(II-14)}$$

$$\phi_n(t) = f_1(t)^{*[a1]} * f_2(t^{*[a2]} * \dots f_j(t)^{*[aj]} * \dots * f_r(t)^{*[q]}, \text{ pour } k+1 > r \quad \text{(II-15)}$$

avec $q = n - \sum_{j=1}^{r-1} a_j$,

L'expression de h(t) est obtenue en remplaçant, dans (II-6), $\phi_n(t)$ par les expressions (II-14) et (II-15) et en passant par la transformée de Laplace.

Pour r = 2

$$h(t) = \lambda_2 + \left\{ (\lambda_1 - \lambda_2)\, \lambda_1^{a_1-1}\, \frac{t^{a_1-1}}{(a_1-1)!} + \sum_{k=1}^{a_1-1} \lambda_1^{k-1}\, \frac{t^{k-1}}{(k-1)!} \right\} \exp-(\lambda_1 t) \quad \text{(II-16)}$$

Pour r > 2

$$h(t) = \lambda_r + \sum_{i=1}^{r-1} \beta_i(t) \exp(-\lambda_i t) \quad \text{(II-17)}$$

$$\beta_i(t) = \lambda_i \left\{ \prod_{j=1}^{i-1} \left(\frac{\lambda_j}{\lambda_j - \lambda_i} \right)^{a_j} \sum_{k=1}^{a_i} \frac{(\lambda_i t)^{k-1}}{(k-1)!} \right\}$$

$$+ \left\{ \sum_{m=i+1}^{r-2} \prod_{j=1;\, j\neq i}^{m-1} \left(\frac{\lambda_j}{\lambda_j - \lambda_i} \right)^{a_j} \sum_{k=1}^{a_m} \left(\frac{\lambda_m}{\lambda_m - \lambda_i} \right)^{k} \right.$$

$$+ \prod_{j=1}^{r-2} \left(\frac{\lambda_j}{\lambda_j - \lambda_{r-1}} \right)^{a_j} \sum_{k=1}^{a_{r-1}-1} \left(\frac{\lambda_{r-1}}{\lambda_{r-1} - \lambda_i} \right)^{k}$$

$$\left. + \left(\frac{\lambda_i - \lambda_r}{\lambda_i} \right) \prod_{j=1;\, j\neq i}^{r-1} \left(\frac{\lambda_j}{\lambda_j - \lambda_i} \right)^{a_j} \right\} \lambda_i^{a_i}\, \frac{t^{a_i-1}}{(a_i-1)!} \quad \text{(II-18)}$$

et $\prod_{j=a}^{b} x_j = 1$ pour $b < a$

On peut vérifier que dans le cas où tous les a_j sont égaux à 1, les expressions (II-16) à (II-18) correspondent bien à celles données par (II-7) à (II-9) respectivement.

Pour qu'il y ait croissance de fiabilité, il faut que h(t) soit une fonction non croissante du temps, ceci est réalisé si les λ_i sont tels que :

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{r-1} \geq \lambda_r \qquad \text{(II-19)}$$

De même que précédemment, si les λ_i sont croissants puis décroissants h(t) suit leur évolution.

On peut remarquer que plus les a_j sont faibles, plus rapide est la croissance de la fiabilité (influence des termes en t^{a_i-1} qui atténue la décroissance de l'exponentielle), l'optimum correspondant au cas où les corrections ont lieu avant la reprise des traitements (a_i=1). Les courbes de la figure II-4 illustrent ce point.

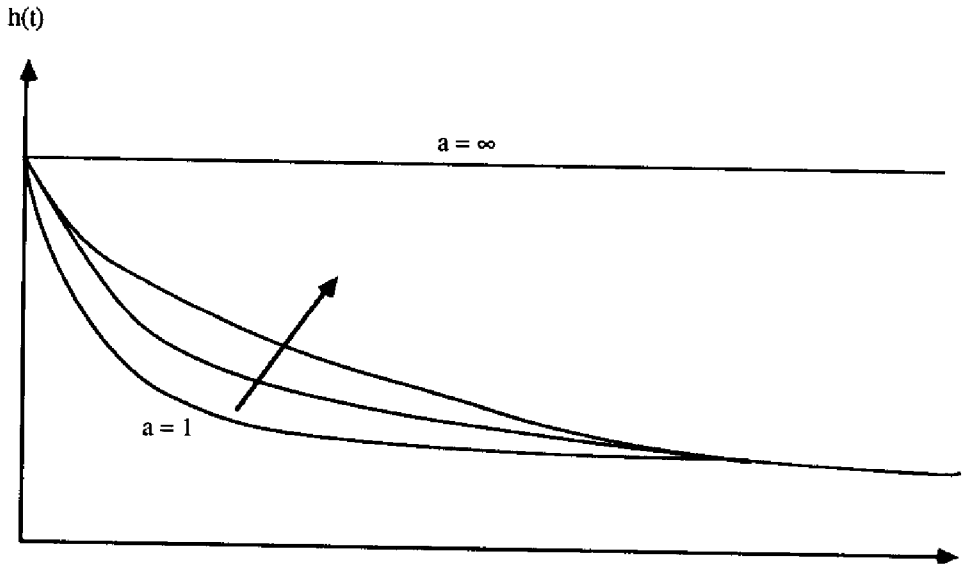


Figure II-4 : Influence du nombre a de défaillances avant correction sur l'intensité de défaillance.

II-1-1-3-2 Cas où l'introduction d'une édition est indépendante des défaillances

Si on considère un logiciel ayant une seule instance, les défaillances sont toutes observées sur cette même instance et les corrections sont introduites progressivement sur cette instance.

Si on considère un logiciel en opération, avec N instances sur des sites différents ou non, le logiciel est corrigé après occurrence de a_j défaillances sur ces N sites. Etant donné que les défaillances peuvent avoir lieu sur un grand nombre d'instances, il peut y avoir introduction d'une nouvelle édition sur un site particulier alors qu'aucune défaillance n'a eu lieu pour ce site. Soit b_j le nombre moyen de défaillances ayant eu lieu sur un site avant introduction d'une nouvelle édition, b_j est donné par : $b_j = \frac{a_j}{N}$,

- pour $a_j \geq N$, on a $b_j \geq 1$: la fréquence des corrections est alors inférieure à celle des défaillances, ce cas est analogue au cas d'une seule instance,
- pour $a_j < N$, b_j est inférieur à 1, b_j étant défini par rapport à un utilisateur "moyen", il peut être inférieur à 1 ; ceci signifie que pour un site moyen, il y a plus d'introductions d'éditions que d'occurrences de défaillances ; de façon plus précise : pour le(s) site(s) où la défaillance a eu lieu il est forcément égal à 1 puisqu'il s'agit de maintenance corrective ; pour les autres sites, l'introduction de ou des corrections constitue de la **maintenance préventive** afin d'éviter les redécouvertes. Ce cas est caractérisé par le fait que la fréquence des corrections pour un site moyen est supérieure à celle des défaillances.

Afin d'illustrer le rapport entre la fréquence de défaillance et la fréquence de correction, nous considérons les autocommutateurs téléphoniques étudiés dans le chapitre V en vie opérationnelle stabilisée :

- pour l'E-10 : il y a eu pendantla dernière année 5 défaillances et 3 corrections sur l'ensemble des 1400 autocommutateurs (sites) en service , ce qui signifie que la plupart des sites ont eu 3 changements d'édition alors qu'ils n'ont eu aucune défaillance,
- pour le TROPICO-R on a observé les 6 derniers mois 7 défaillances-corrections sur l'ensemble des 15 sites installés, ce qui signifie que certains sites ont subi 7 changements d'éditions alors qu'ils n'ont eu aucune défaillance.

Pour ces deux systèmes la fréquence des corrections est supérieure à celle des défaillances.

On va considérer successivement le cas où la fréquence des corrections pour un site moyen est inférieure à celle des défaillances et le cas où elle est supérieure à cette fréquence.

Cas où la fréquence des corrections sur un site moyen est inférieure à celle des défaillances

Dans ce cas il faut tenir compte du temps écoulé entre la dernière défaillance correspondant à l'édition j et l'introduction de l'édition j+1, soit T_j ce temps (j = 1, ..., k) et $g_j(t)$ la fonction densité de probabilité associée.
La relation entre les différents temps est schématisée à la figure II-5.

Les relations (II-11) à (II-13) sont inchangées, (II-10) devient :

$$Y_j = T_j + \sum_{i=1}^{a_j} T_{j,i} \,, \qquad \text{(II-20)}$$

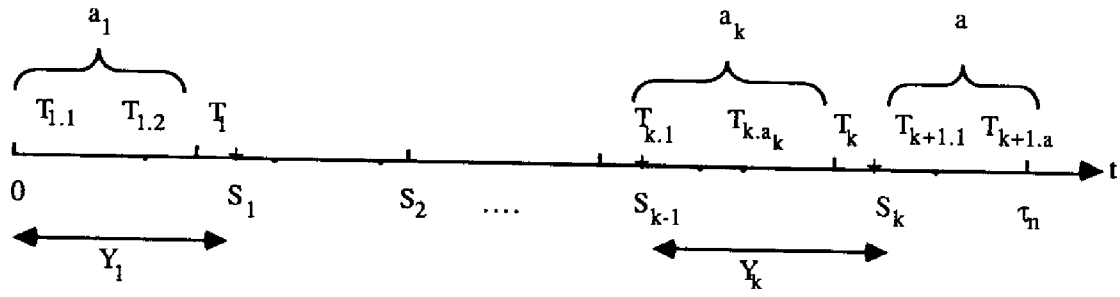


Figure II-5 : Relation entre les différents intervalles de temps.

ce qui donne pour $\phi_n(t)$:

$$\phi_n(t) = f_1(t)^{*[a1]} * g_1(t) * f_2(t)^{*[a2]} * g_2(t) * \ldots f_j(t)^{*[aj]} * g_j(t) * \ldots * f_{k+1}(t)^{*[a]}, \quad \text{pour } k+1 \le r \quad \text{(II-21)}$$

$$\phi_n(t) = f_1(t)^{*[a1]} * g_1(t) * f_2(t)^{*[a2]} * g_2(t) * \ldots f_j(t)^{*[aj]} * g_j(t) * \ldots * f_r(t)^{*[q]}, \quad \text{pour } k+1 > r \quad \text{(II-22)}$$

avec $q = n - \sum_{j=1}^{r-1} a_j$,

Il est nécessaire d'avoir les distributions des T_j pour obtenir l'expression de l'intensité de défaillance en utilisant (II-6).

Le temps T_j correspond à l'intervalle de temps entre l'instant d'occurrence de la dernière défaillance (pour une édition donnée) et l'instant où la ou les corrections des fautes qui ont été activées sont introduites sous forme d'une nouvelle édition. Ce temps est étroitement lié à la nature de la faute qui a été activée, à la conséquence de la défaillance, à la disponibilité des personnes chargées des corrections ... ; dans la méconnaissance de ce processus et par commodité de calcul, nous prendrons une distribution exponentielle indépendante de l'ordre des défaillances.

Dans le cas où les t_j ont une distribution exponentielle avec un taux identique v, l'expression de h(t) est donnée par les expressions (II-23) à (II-27).

Pour r = 2

$$h(t) = \lambda_2 + \left(\lambda_2 \ \lambda_1^{a_1-1} \ \frac{\nu}{\lambda_1-\nu} \ \frac{t^{a_1-1}}{(a_1-1)!} + \sum_{k=1}^{a_1} \lambda_1^{k} \ \frac{t^{k-1}}{(k-1)!} \right) \exp-(\lambda_1 t)$$

$$+ \left(\frac{\lambda_1}{\lambda_1-\nu} \right)^{a_1} \lambda_2 \exp(-\nu t) \qquad \text{(II–23)}$$

Pour r > 2

$$h(t) = \lambda_r + \sum_{i=1}^{r-1} \gamma_i(t) \exp(-\lambda_i t) + \sum_{k=2}^{r} \delta_k \exp(-\nu t) \qquad \text{(II–24)}$$

avec :

$$\gamma_i(t) = \lambda_i \left\{ \left(\frac{\nu}{\nu-\lambda_i} \right)^{i-1} \prod_{j=1}^{i-1} \left(\frac{\lambda_j}{\lambda_j-\lambda_i} \right)^{a_j} \ \sum_{k=1}^{a_i} \frac{(\lambda_i t)^{k-1}}{(k-1)!} \right\}$$

$$+ \left\{ \sum_{m=i+1}^{r-1} \left(\frac{\nu}{\nu-\lambda_i} \right)^{i-1} \prod_{j=1;j\neq i}^{m-1} \left(\frac{\lambda_j}{\lambda_j-\lambda_i} \right)^{a_j} \ \sum_{k=1}^{a_m} \left(\frac{\lambda_m}{\lambda_m-\lambda_i} \right)^{k} \right.$$

$$\left. + \prod_{j=1}^{r-1} \left(\frac{\lambda_j}{\lambda_j-\lambda_{r-1}} \right)^{a_j} \left(\frac{\nu}{\nu-\lambda_{r-1}} \right)^{r-1} \left(-\frac{\lambda_r}{\lambda_{r-1}} \right) \right\} \lambda_i^{a_i} \frac{t^{a_i-1}}{(a_i-1)!} \qquad \text{(II–25)}$$

et $\prod_{j=a}^{b} x_j = 1$ pour $b < a$

$$\delta_k = \nu^{k-1} \prod_{j=1}^{k-1} \left(\frac{\lambda_j}{\lambda_j-\nu} \right)^{a_j} \sum_{j=1}^{a_k} \left(\frac{\lambda_k}{\lambda_k-\nu} \right)^{j} \frac{t^{k-2}}{(k-2)!} \quad 2 \le k \le r-1 \qquad \text{(II–26)}$$

$$\delta_r = -\lambda_r \nu^{r-2} \prod_{j=1}^{r-1} \left(\frac{\lambda_j}{\lambda_j-\nu} \right)^{a_j} \frac{t^{r-2}}{(r-2)!} \qquad \text{(II–27)}$$

Le fait de prendre la même distribution pour tous les t_j n'enlève rien à la généralité du problème mais simplifie l'expression de h(t) ; de même la prise en compte d'une distribution non exponentielle compliquerait d'avantage l'expression de h(t).

Pour $\nu = \infty$ on retrouve les expressions (II-16) à (II-18) correspondant au cas où l'introduction d'une édition coïncide avec une défaillance.

Pour $\nu \neq \infty$, ν a le même effet qu'une augmentation des a_j : ceci est dû à l'influence des termes en exp(-νt) qui entraîne une légère augmentation de h(t), donc un ralentissement de la croissance de fiabilité. Ce résultat est tout à fait logique et prévisible : *la correction différée entraîne une croissance de fiabilité moins rapide que la correction immédiate, au même titre qu'une politique de correction par lot.*

Cas où la fréquence des corrections sur un site moyen est supérieure à celle des défaillances

Localement l'évolution de la fiabilité du logiciel est beaucoup plus influencée par la fréquence des changements d'édition que par les défaillances ayant lieu sur cette version puisque la fréquence des corrections est plus élevée que celle des défaillances. L'évolution de la fiabilité sera donc étudiée au travers de l'étude de la variable aléatoire intervalle de temps entre deux corrections, soit t_j ce temps ; t_j correspond à la durée de la j ième édition.

Soit r le numéro de l'édition à partir de laquelle le logiciel atteint un comportement stable. On suppose qu'au delà de r le taux de défaillance du logiciel est constant soit parce qu'on n'introduit plus de nouvelle édition soit parce que la différence entre les taux de défaillance de l'ancienne et de la nouvelle édition n'est pas significative.

L'indice j est défini comme suit :

j<r : nombre de reprises des traitements suite à l'introduction d'une nouvelle édition,

j>r : nombre de reprises des traitements soit suite à une défaillance sans changement d'édition soit suite à l'introduction d'une nouvelle édition sans changement significatif du taux de défaillance résiduel.

Soit $g_j(t)$ (j<r) la densité de probabilité de la variable aléatoire durée de la j ième édition, et $1/\nu_j$ le premier moment de cette distribution. Deux questions se posent alors : la nature de cette distribution et le sens de variation du premier moment. En fait, la distribution du temps entre introduction d'éditions est conditionnée par celle des intervalles entre défaillances, nous prendrons une distribution exponentielle, le taux correspondant étant égal à ν_j. Le sens de variation des ν_j est aussi conditionné par celui des taux de défaillances : si les défaillances sont de plus en plus rapprochées, les corrections le sont aussi et au contraire des défaillances de plus en plus espacées entraînent des corrections de moins en moins fréquentes.

Soit ν_r le taux de relance du logiciel après introduction de l'édition r. Si aucune édition n'est introduite au delà de r, ν_r correspond alors au taux de défaillance résiduel λ_r relatif à un site "moyen".

Si n représente le nombre total de reprises des traitements, τ_n correspond alors à l'instant de reprise des traitements et $\phi_n(t)$ est sa distribution.

L'établissement de l'expression de h(t) se fait de la même manière que pour le cas où $a_j = 1, j = 1, ..., r-1$, en remplaçant $f_j(t)$ par $g_j(t)$ (ce qui revient à remplacer λ_j par ν_j dans les expressions (II-7) à (II-9)). Ce qui donne :

Pour r = 2

$$h(t) = \nu_2 + (\nu_1 - \nu_2) \exp -(\nu_1 t) \qquad \text{(II-28)}$$

Pour r > 2

$$h(t) = \nu_r + \sum_{i=1}^{r-1} \alpha'_i \exp(-\nu_i t) \qquad \text{(II-29)}$$

avec :

$$\alpha'_i = \left[\sum_{k=i}^{r-2} \nu_i \prod_{j=1;\ j\neq i}^{k} \frac{\nu_j}{\nu_j - \nu_i} \right] + (\nu_i - \nu_r) \prod_{j=1;\ j\neq i}^{k} \frac{\nu_j}{\nu_j - \nu_i} \qquad \text{(II-30)}$$

et $\prod_{j=a}^{b} x_j = 1$ pour $b < a$

Tenant compte du fait que $\lambda_j \leq \nu_j$, et dans le cas où (II-19) est vérifiée pour les ν_j (décroissance stochastique des taux de changement d'éditions ou de relance), la courbe intensité de changement d'éditions se situe au dessous des courbes intensité de défaillance correspondant à $a_j = 1, j=1, ..., r-1$ (figure II-4) : la croissance de fiabilité est beaucoup plus rapide dans ce cas que dans les cas précédents.

Ces résultats, en accord avec ceux publiés dans [Bak 88], montrent l'influence positive d'un grand nombre d'instances en vie opérationnelle sur l'évolution de la fiabilité d'un logiciel : la maintenance préventive sur certains sites entraîne une croissance de fiabilité plus forte que celle résultant d'une maintenance corrective sur les sites concernés uniquement.

Les recommandations données dans [Gra 85] concernant la maintenance préventive sont plus nuancées : les résultats d'une étude effectuée par Tandem montrent effectivement qu'un grand pourcentage de défaillances est dû à des fautes de logiciel et de matériel connues dont les corrections sont disponibles et non encore installées. L'auteur constate que ceci correspond plus au cas du matériel et que pour le logiciel il recommande d'effectuer la maintenance préventive uniquement pour les fautes qui entraînent la défaillance du

système. Pour les autres types de fautes il vaut mieux attendre une édition "majeure" du logiciel bien testée. En fait notre résultat est en accord avec ces recommandations puisque nous ne distinguons pas ici les conséquences des défaillances sur le service rendu par le logiciel.

II-1-1-4 *Changement de spécifications*

Au niveau du programme, un changement de spécifications se traduit par une modification des sous-espaces E_e et E_{af} ainsi que du taux de défaillance.

Plusieurs cas sont à considérer :

- s'il n'y a pas d'éditions intermédiaires entre deux versions (corrections introduites par lots en même temps que l'introduction d'une version), on se trouve dans des situations analogues à celles qui viennent d'être analysées en remplaçant édition par version,

- s'il y a des éditions intermédiaires :
  - si l'augmentation du taux de défaillance due au changement des spécifications est compensée par les corrections introduites en même temps, le taux de défaillance résultant peut décroître,
  - si le changement de spécifications entraîne une augmentation du taux de défaillance correspondant, on aura un taux de défaillance décroissant pour une version donnée qui croît brusquement au moment de l'introduction de la nouvelle version pour décroître à nouveau ; les modèles de comportement que nous venons d'établir étant basés directement sur les valeurs des taux de défaillance, sont tout à fait utilisables dans ce cas aussi.

On voit que quel que soit le cas de figure (avec ou sans éditions intermédiaires), les modèles précédents restent valables même avec changements de version, il faut cependant tenir compte de la variation du taux de défaillance au moment de l'introduction de la nouvelle version.

II-1-2 Modèles de disponibilité

L'évaluation de la disponibilité nécessite la connaissance des deux variables aléatoires : intervalle entre défaillances et durée des arrêts. Pour le processus de défaillances nous considérons les mêmes hypothèses que dans les cas précédents : processus de Poisson par morceaux, et on suppose qu'au bout d'un certain nombre de corrections le taux de défaillance atteint une valeur constante non nulle.

Comme dans le cas de la fiabilité, plusieurs politiques de corrections sont à considérer, nous traiterons d'abord en détail le cas de la correction immédiate et nous aborderons ensuite le cas de la correction différée.

II-1-2-1 *Correction immédiate*

Soient T_n la variable aléatoire intervalle entre les défaillances n-1 et n, et Y_n la variable aléatoire durée de correction du logiciel après la défaillance n. Les fonctions de répartition correspondantes sont respectivement $F_n(t)$ et $G_n(t)$ et les fonctions densité de probabilité sont respectivement $f_n(t)$ et $g_n(t)$. X_n correspond à la durée de fonctionnement continu après la reprise des traitements suite à la défaillance n-1.
Soient S_n les instants de renouvellement (les instants où le logiciel est relancé après défaillance) ; notons H(t) et h(t) la fonction de répartition et la fonction densité de probabilité qui lui sont associées.
Soit $Z_n = T_n + Y_n$; les fonctions de répartition et de densité de probabilité de Z_n sont notées respectivement $\Phi_n(t)$ et $\phi_n(t)$.
La succession de ces différents temps est donnée à la figure II-6.

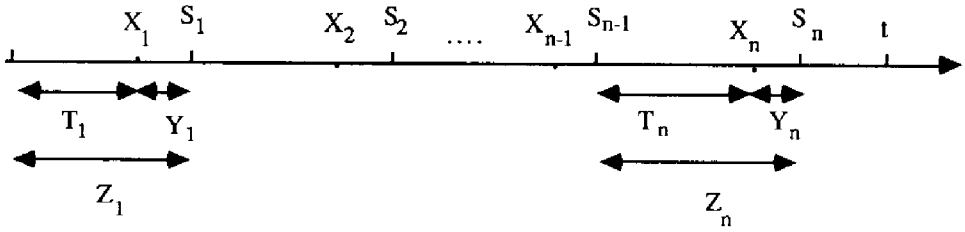


Figure II-6 : Intervalles entre défaillances et durées des corrections

La disponibilité instantanée A(t) est définie par :

A(t) = P{ le logiciel délivre un service approprié à l'instant t }.

Considérons l'événement E_n tel qu'à l'instant S_n il s'est produit n défaillances et qu'à l'instant t le logiciel délivre un service approprié.

$$E_n = \{ S_n < t < S_n + T_{n+1} \}$$

La disponibilité est alors donnée par :

$$A(t) = P \{ E_0 \, U \, E_1 \, U \ldots E_n \, U \, E_{n+1} \, U \ldots \}$$

$$A(t) = P \{ \underset{n=0,\infty}{U} E_n \}$$

Les événements E_n étant incompatibles, il vient :

$$A(t) = \sum_{n=0}^{\infty} P \{ E_n \}$$

Calcul de $P \{ E_n \}$

$$P \{ E_0 \} = 1 - F(t) = R_1(t).$$

$$P \{ E_n \} = P \{ S_n < t < S_n + T_{n+1} \} \qquad n = 1, 2, \ldots, \infty$$

$$= \int_0^t P\{ x < S_n < x+dx \} \, . \, P\{ T_{n+1} > t-x \}$$

$$= \int_0^t d \, \Phi_n(x) \, [1 - F_{n+1}(t-x)]$$

$$= \int_0^t \phi_n(x) \, R_{n+1}(t-x) \, dx$$

$$= \phi_n(t) * R_{n+1}(t) \qquad n = 1, 2, \ldots, \infty$$

où $R_n(t)$ est la fonction de fiabilité associée à la défaillance numéro n et le symbole * représente l'opération de convolution.

Ce qui donne :

$$A(t) = \sum_{n=0}^{\infty} P \{ E_n \} = R_1(t) + \sum_{n=1}^{\infty} \phi_n(t) * R_{n+1}(t) \qquad \text{(II-31)}$$

L'expression de l'intensité de défaillance s'obtient en passant par la transformée de Laplace. Notons $\tilde{B}(s)$ la transformée de Laplace de la fonction B(t). La transformée de Laplace de A(t) est alors donnée par :

$$\tilde{A}(s) = \tilde{R}_1(s) + \sum_{n=1}^{\infty} \tilde{\phi}_n(s)\ \tilde{R}_{n+1}(s)$$

En supposant qu'après r défaillances le logiciel atteint un comportement stable (régime stationnaire), on peut faire l'hypothèse qu'à partir de l'indice r les fonctions densité de probabilité et de répartition pour les deux variables aléatoires (intervalle entre défaillances et durée des corrections) sont inchangées.

Les $\tilde{\phi}_n(s)$ ont alors pour expression :

$$\begin{aligned} \tilde{\phi}_n(s) &= \tilde{f}_1(s)\,\tilde{g}_1(s)\ \tilde{f}_2(s)\,\tilde{g}_2(s)\ \ldots\ \tilde{f}_n(s)\,\tilde{g}_n(s) && \text{pour } n \le r \\ \tilde{\phi}_n(s) &= \tilde{f}_1(s)\,\tilde{g}_1(s)\ \tilde{f}_2(s)\,\tilde{g}_2(s)\ \ldots\ \tilde{f}_r(s)^{[n-r+1]}\,\tilde{g}_r(s)^{[n-r+1]} && \text{pour } n > r \end{aligned} \qquad \text{(II-32)}$$

avec :

$$\tilde{f}_n(s) = \frac{\lambda_n}{s+\lambda_n} \qquad \text{et} \qquad \tilde{R}_n(s) = \frac{1}{s} - \frac{1}{s}\,\frac{\lambda_n}{s+\lambda_n} = \frac{1}{s+\lambda_n}$$

Nous supposons dans un premier temps que les taux de correction sont exponentiellement distribués, ce qui donne :

$$\tilde{g}_n(s) = \frac{\mu_n}{s+\mu_n}$$

Pour l'évaluation de l'indisponibilité, soit on évalue $\bar{A} = 1-A(t)$, soit on l'évalue directement en considérant les événements D_n tels qu'à l'instant X_n il s'est produit n défaillances et qu'à l'instant t le logiciel est défaillant.

Il vient :

$$\tilde{\bar{A}}(s) = \frac{\lambda_r \prod_{n=1}^{r-1} \lambda_n \mu_n + s\,(s+\lambda_r+\mu_r) \sum_{j=1}^{r-1} \lambda_j \prod_{n=1}^{j-1} \lambda_n \mu_n \prod_{n=j+1}^{r-1} (s+\lambda_n)(s+\mu_n)}{s\,(s+\lambda_r+\mu_r) \prod_{n=1}^{r-1} (s+\lambda_n)(s+\mu_n)} \qquad \text{(II-33)}$$

Le comportement stabilisé est obtenu à l'aide de :

$$\lim_{t\to\infty} \bar{A}(t) = \lim_{s\to 0} s\,\tilde{\bar{A}}(s) = \frac{\lambda_r}{\lambda_r + \mu_r} \approx \frac{\lambda_r}{\mu_r} = \bar{A}$$

Ce résultat est tout à fait logique car si on considère un système à taux de défaillance et de restauration constants, l'indisponibilité est égale (au premier ordre) au rapport du taux de défaillance au taux de correction.

L'expression de $\bar{A}(t)$ s'obtient en passant par la transformée de Laplace inverse et en effectuant un développement limité en $\frac{\lambda_n}{\mu_n} << 1$, soit :

$$\bar{A}(t) = \frac{\lambda_r}{\mu_r} + \sum_{n=1}^{r-1} \alpha_n \exp(-\lambda_n t) - \frac{\lambda_1}{\mu_1}\exp(-\mu_1 t) \qquad \text{(II-34)}$$

avec :

$$\alpha_n = \sum_{j=n}^{r-1} \frac{\lambda_j}{\mu_j} \frac{\prod_{i=1}^{j-1} \lambda_i}{\prod_{i=1, i\neq n}^{j} (\lambda_i - \lambda_n)} - \frac{\lambda_r}{\mu_r} \prod_{j=1,\, j\neq n}^{r-1} \frac{\lambda_j}{\lambda_j - \lambda_n} \qquad \text{(II-35)}$$

$\bar{A}(0) = 0$ implique la relation suivante entre les α_n :

$$\sum_{n=1}^{r-1} \alpha_n = \frac{\lambda_1}{\mu_1} - \frac{\lambda_r}{\mu_r}, \qquad \text{(II-36)}$$

relation qui peut être vérifiée directement à partir de l'expression des α_n.

Ces résultats sont identiques à ceux obtenus dans [Lap 83] à l'aide d'une formulation markovienne.

Résultats et Commentaires :

Influence des taux de défaillance :

Dans le cas où il y a croissance monotone de fiabilité (taux de défaillance décroissants), et que les taux de corrections sont tels que leur possible décroissance ne compense pas la décroissance des taux de défaillance :

$$\frac{\lambda_{n+1}}{\mu_{n+1}} \leq \frac{\lambda_n}{\mu_n}$$

la courbe d'indisponibilité a un seul maximum et possède les propriétés suivantes :

- La valeur maximale de l'indisponibilité est essentiellement fonction de λ_1 : une valeur approchée peut être obtenue en remarquant que le dépassement est maximum lorsque le terme en $\exp(-\mu_1 t)$ est devenu négligeable alors que les termes en $\exp(-\lambda_n t)$ n'ont pas encore significativement décru. Il vient alors :

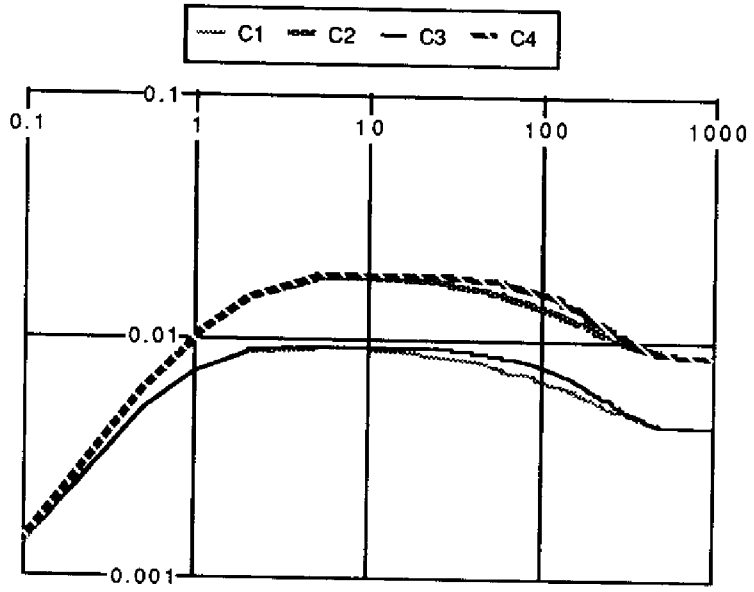
$$\bar{A}_{max} \simeq \sum_{n=1}^{r-1} \alpha_n + \frac{\lambda_r}{\mu_r} = \frac{\lambda_1}{\mu_1}$$

- Le temps de montée de l'indisponibilité est de l'ordre de quelques $\frac{1}{\mu_1}$.
- Les taux λ_n (n = 2,...,r-1) influencent la largeur du palier.
- λ_r donne directement la valeur asymptotique.

La figure II-7 permet d'illustrer les propriétés citées ci-dessus ; elle donne les courbes d'indisponibilité pour des λ_n décroissants et en supposant $\mu_n = \mu$, $\forall$ n. Les valeurs choisies pour les différents λ_n sont prises à titre indicatif. Bien que ces courbes soient tracées pour r=4 afin de simplifier l'expression de $\bar{A}(t)$, elles sont beaucoup plus générales et leur allure correspond au cas de croissance monotone de fiabilité.

Ce qui précède montre que l'indisponibilité asymptotique n'est pas représentative du comportement du logiciel durant une période relativement longue : le régime stable est atteint lorsque les termes en $\exp(-\lambda_n t)$ deviennent négligeables, soit au bout de quelques $\frac{1}{\lambda_1}$, disons à partir de : $t \simeq 10 \frac{1}{\lambda_1}$, ($e^{-10} = 4{,}5\ 10^{-5}$).

Une augmentation du taux de défaillance (décroissance de fiabilité) peut entraîner l'existence de maxima locaux et donc d'une fluctuation de la courbe d'indisponibilité avant que le régime stable ne soit atteint.



Courbe	λ_1	λ_2	λ_3	λ_4	μ
C1	$1,5\ 10^{-2}$	$1,1\ 10^{-2}$	$9\ 10^{-3}$	$7\ 10^{-3}$	1,6
C2	$1,5\ 10^{-2}$	$1,1\ 10^{-2}$	$9\ 10^{-3}$	$7\ 10^{-3}$	0,8
C3	$1,5\ 10^{-2}$	$1,4\ 10^{-2}$	$1,2\ 10^{-3}$	$7\ 10^{-3}$	1,6
C4	$1,5\ 10^{-2}$	$1,4\ 10^{-2}$	$1,2\ 10^{-3}$	$7\ 10^{-3}$	0,8

Figure II-7 : Courbes d'indisponibilité $\bar{A}(t)$: influence de μ, λ_2 et λ_3.
(échelle Log-Log)

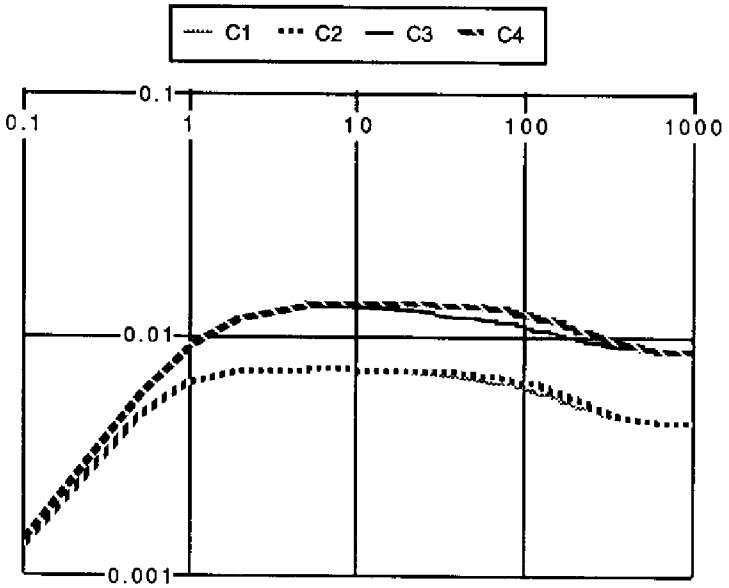
Influence des durées de correction :

En se plaçant dans le cas de croissance monotone de fiabilité, l'influence des durées de correction est la suivante :

- μ_1 fixe le maximum et le temps de montée de l'indisponibilité.
- Les μ_n (n = 2,...,r-1) influencent la largeur du palier : une décroissance des μ_n a un effet analogue à celui entraîné par une croissance des λ_n, toutefois l'influence des μ_n est moins importante que celle des λ_n puisqu'ils n'interviennent qu'au niveau de l'expression des α_n ; une forte décroissance peut entraîner des maxima locaux.
- μ_r donne directement la valeur asymptotique.

La figure II-8 illustre ces propriétés.

Concernant le sens de variation des durées de correction ($\frac{1}{\mu_n}$) tout au long du cycle de vie, deux raisonnements contradictoires peuvent mener à des conclusions opposées :



Courbe	μ_1	μ_2	μ_3	μ_4
C1	2	1,8	1,7	1,6
C2	2	1,6	1,6	1,6
C3	1,1	1	0.9	0.8
C4	1,1	0,8	0,8	0,8

$\lambda_1 = 1{,}5\ 10^{-2}$	$\lambda_2 = 1{,}1\ 10^{-2}$	$\lambda_3 = 9\ 10^{-3}$	$\lambda_4 = 7\ 10^{-3}$

Figure II-8 : Courbes d'indisponibilité $\bar{A}(t)$: influence des variations de μ_n.

- on peut supposer que les premières erreurs sont plus faciles à corriger et nécessitent ainsi des temps de correction plus courts que les dernières, ceci semble vérifié aussi bien en développement [Basi 88] qu'en vie opérationnelle,
- on peut, au contraire, espérer que les programmeurs deviennent de plus en plus familiers avec le programme et mettent moins de temps pour corriger les erreurs tardives.

Ces deux effets peuvent aussi se compenser au cours du temps. Le manque de données d'expérience concernant les durées de correction ne permet pas de dégager de relation liant les taux de correction successifs, notamment en vie opérationnelle.

L'étude de l'influence d'une distribution non exponentielle des temps de correction a été effectuée dans [Cos 78] et montre que tant que les durées de corrections sont faibles par rapport aux intervalles entre défaillances ($\lambda_n << \mu_n$), un temps de correction déterministe n'a pas d'influence significative pour : $t > \frac{1}{\mu_1}$.

II-1-2-2 *Correction différée*

Après défaillance du logiciel, il y a réinitialisation du système ; selon les conséquences de la défaillance sur l'état du logiciel la réinitialisation peut consister à relancer le logiciel si l'état du logiciel n'a pas été affecté, si par contre la défaillance a affecté l'intégrité du logiciel (effacement d'une partie des données nécessaires à la relance, modifications irréversibles par simple relance) il est nécessaire de recharger le logiciel. La durée du chargement risque d'être plus longue que la durée de relance et il faut tenir compte de toutes ces causes d'arrêt. On se place dans les mêmes conditions que celles du paragraphe II-1-1-3 :

- taux de défaillance inchangé après réinitialisation du système (soit relance soit rechargement),
- variation du taux de défaillance après introduction d'une nouvelle édition,
- l'édition i est introduite après occurrence de a_i défaillances.

La fonction densité de probabilité associée à la i ième édition est notée comme précédemment $f_i(t)$ et la fonction densité de probabilité associée à la durée de la i ième correction est notée comme précédemment $g_i(t)$.

Soient r le numéro de l'édition à partir de laquelle le logiciel atteint un comportement stable et k le nombre total d'éditions introduites jusqu'à l'instant t.

Si on suppose que les temps d'arrêt pour réinitialisation sont négligeables l'expression de la disponibilité est établie à partir de (II-32) en remplaçant les $\tilde{\phi}_n(s)$ par :

$$\tilde{\phi}_n(s) = \tilde{f}_1(s)^{[a_1]}\,\tilde{g}_1(s)\quad \tilde{f}_2(s)^{[a_2]}\,\tilde{g}_2(s)\ \ldots\ \tilde{f}_n(s)^{[a_n]}\,\tilde{g}_n(s) \qquad \text{pour } n \le r$$

$$\tilde{\phi}_n(s) = \tilde{f}_1(s)^{[a_1]}\,\tilde{g}_1(s)\quad \tilde{f}_2(s)^{[a_2]}\,\tilde{g}_2(s)\ \ldots\ \tilde{f}_r(s)^{[q]}\,\tilde{g}_r(s)^{[p]} \qquad \text{pour } n > r$$

avec $q = n - \sum_{j=1}^{r-1} a_j$ et $p = k-r$,

Dans le cas où les temps d'arrêt pour réinitialisation ne peuvent être négligés, les évaluations effectuées dans le paragraphe précédent restent valables en distinguant les taux d'arrêt pour réinitialisation des taux d'arrêt pour correction.

Les termes $\tilde{f}_n(s)^{[a_n]}$ font apparaître le facteur t^{a_n-1} au niveau des paramètres α_n de l'expression de $\bar{A}(t)$: (II-34) et (II-35). Ces facteurs entraînent une décroissance de l'indisponibilité moins rapide qu'en cas de correction immédiate.

II-1-3 Conclusions

Les évaluations de l'intensité de défaillance et de la disponibilité effectuées dans ce paragraphe supposent que λ_i et a_i sont deux variables connues a priori afin de bien voir l'influence de chacune d'elles séparément. On peut remarquer que, malgré ces hypothèses simplificatrices, quelle que soit la politique de correction adoptée, les expressions des mesures de la sûreté de fonctionnement sont très complexes et pratiquement inutilisables pour des cas réels ; elles permettent cependant de voir l'évolution du logiciel dans des conditions particulières.

Nous pensons également nécessaire d'insister sur le fait que les cas que nous avons modélisés ne sont que des cas très particuliers volontairement simplifiés et que dans la pratique on se trouve généralement confronté à des situations beaucoup plus complexes. En effet, le nombre de relances avant correction (a_i) n'est pas connu a priori, il dépend de facteurs de natures diverses : la nature même de la faute, l'environnement d'utilisation, la conséquence de la défaillance correspondante sur le service délivré par le système, la disponibilité du personnel assurant les corrections, la pression exercée par les utilisateurs sur le personnel de maintenance, la nature du contrat de maintenance... Tous ces facteurs font que l'idéal serait de considérer ce nombre comme étant lui même une variable aléatoire. On pourrait aussi considérer dans un deuxième temps λ comme une variable aléatoire (discrète ou continue) aussi. Le problème serait alors de déterminer les distributions de ces variables aléatoires. Ceci compliquerait davantage les expressions des différentes mesures sans grand intérêt d'un point de vue pratique.

Notre raisonnement est le suivant : les modèles que nous venons d'établir permettent de bien comprendre les phénomènes mis en jeu :
- succession des éditions et des versions,
- influence d'un grand nombre d'instances,
- influence de différentes politiques de correction,
- influence des taux de correction sur l'indisponibilité.

Ces modèles constituent ce qui est couramment appelé **des modèles de connaissance**.

La solution que nous proposons consiste à approcher les courbes (intensité de défaillance et indisponibilité) par des courbes traduisant la même croissance (ou décroissance) de la sûreté de fonctionnement avec des expressions plus simples, donc plus maniables et à utilisation plus souple : ces modèles de connaissance sont ainsi approchés par des **modèles d'action**. Les modèles d'action sont paramètrés et les mesures de la sûreté de

fonctionnement sont approchées via l'ajustement des paramètres correspondants. Les modèles d'action font l'objet du paragraphe suivant.

II-2 MODELES D'ACTION

Mis à part le modèle hyperexponentiel [Lap 84a et b, Kan 85], tous les modèles d'action qui ont été développés dans la littérature sont des modèles de croissance de fiabilité et ne permettent pas de rendre compte de la croissance de disponibilité. Nous commencerons donc par considérer quelques modèles de croissance de fiabilité et nous développerons ensuite le modèle hyperexponentiel.

Concernant les modèles de fiabilité, les modèles de connaissance que nous avons établis dans le paragraphe précédent sont basés sur deux hypothèses : indépendance des variables aléatoires intervalle entre défaillances et distribution exponentielle de ces variables. Afin d'établir des modèles à expressions plus simples, on introduit des hypothèses supplémentaires. Ces hypothèses consistent à considérer des relations entre ces variables aléatoires.
Nous verrons dans le paragraphe II-2-2 qu'il existe deux classes de modèles parmi lesquelles les modèles basés sur la théorie des **Processus de Poisson Non Homogènes** (abréviation : **NHPP**). De par leur importance aussi bien sur le plan pratique que théorique, nous leur accorderons le paragraphe suivant avant de présenter quelques modèles de croissance de fiabilité.

Ce paragraphe est de ce fait divisé en quatre parties :
- caractérisation des modèles NHPP,
- présentation de modèles de croissance de fiabilité,
- présentation du modèle hyperexponentiel qui permet de modéliser à la fois la croissance de fiabilité et la croissance de disponibilité,
- quelques commentaires sur les modèles.

II-2-1 Modèles basés sur les Processus de Poisson Non Homogènes

Les processus de Poisson non homogènes ont souvent servi de base à la modélisation de la croissance de fiabilité du matériel [Cro 77] et du logiciel [Goe 79, Yam 83, Musa 84, Kan 85...]. Ils permettent en effet d'établir des relations simples entre les différents attributs de la sûreté de fonctionnement précédemment définis (fiabilité, taux de défaillance, intensité de défaillance, nombre cumulé de défaillances...).

II-2-1-1 *Définition*

Un processus stochastique est un NHPP avec une intensité de défaillance h(t) si les relations suivantes sont vérifiées et s'il est à accroissements indépendants [Cox 66] :

$$P\{ N(t+dt) - N(t) = 1 \} = h(t)\, dt$$

$$P\{ N(t+dt) - N(t) \geq 2 \} = o(dt)^2 .$$

Ces relations sont les transcriptions directes des relations (I-4') et (I-5) établies au chapitre I dans le paragraphe I-3-1-2.

Un processus stochastique N(t), $0 < t < \infty$, est à accroissements indépendants [Asc 84] si pour :

$$t_0 < t_1 < \ldots < t_n ,$$

les variables aléatoires : $[N(t_0)], [N(t_1)-N(t_0)], \ldots [N(t_n)-N(t_{n-1})]$

sont indépendantes ; dit en d'autres termes : le nombre d'occurrences du processus sur un intervalle de temps n'est pas influencé par le nombre d'occurrences sur un autre intervalle disjoint.

Bien que cette propriété semble très logique, elle est très difficile à vérifier dans la réalité ; elle permet, cependant, de faciliter considérablement l'expression des différents attributs de la sûreté de fonctionnement. Une étude antérieure [Mil 86] a montré que si le processus étudié n'est pas répétitif (une seule réalisation) il n'est pas possible de distinguer un processus déterministe du processus NHPP ayant les mêmes caractéristiques.

La flexibilité de cette catégorie de modèles provient des faits suivants :

- aucune hypothèse sur les relations entre les différents taux de défaillances successifs n'est établie,
- le numéro de la défaillance n'intervient pas,
- aucune relation n'est supposée entre les processus de défaillance, de relance et de correction et éventuellement de changement de spécification.

Cette dernière propriété permet de considérer qu'un *événement* est soit une défaillance-suivie d'une relance, soit une défaillance-suivie d'une correction, soit l'introduction d'une nouvelle édition ou d'une nouvelle version. Le détail de l'établissement des expressions des mesures de la sûreté de fonctionnements relatives à un NHPP est donné dans l'Annexe 1, seuls les résultats sont résumés ci-après.

II-2-1-2 *Mesures de la sûreté de fonctionnement*

Le nombre d'événements sur un intervalle $[t_1, t_2]$ suit une distribution de Poisson de moyenne égale à :

---

2 $o(t)$ est une fonction qui tend vers zéro quand t tend vers zéro.

$$E[N(t_2)-N(t_1)] = \int_{t_1}^{t_2} h(t)dt = H(t_2)-H(t_1). \qquad \text{(II-37)}$$

Pour $n > n_0$, on a :

$$P[N(t)=n \mid N(t_0)= n_0] = \frac{[H(t)-H(t_0)]^{n-n_0}}{(n-n_0)!}\exp-[(H(t)-H(t_0)], \; t>t_0, \qquad \text{(II-38)}$$

Cette expression permet de travailler sur des données qui sont plus facilement accessibles que les intervalles entre défaillances : le nombre d'événements sur des intervalles déterminés.

Soient :

- s, l'instant d'occurrence du (i-1) ième événement,
- t, le temps courant, avec $t > s$,
- t', le temps écoulé depuis le (i-1) ième événement, $t' = t - s$,

La fonction de survie (ou de fiabilité) associée au i ième événement est donnée par :

$$R_i(t' \mid s) = \exp - [(H(s+t')-H(s)] \qquad \text{(II-39)}$$

Utilisant le fait que la fonction densité de probabilité se déduit de la fonction de fiabilité par :

$$f_i(t' \mid s) = - d[R_i(t' \mid s)] / dt',$$

on obtient :

$$f_i(t' \mid s) = h(s+t') R_i(t' \mid s). \qquad \text{(II-40)}$$

Une propriété très importante des processus de Poisson non homogènes est que le taux de hasard (de défaillance) associé à l'événement i, a la même expression que l'intensité de défaillance, seul l'instant de début d'observation diffère :

$$\lambda_i(t \mid s) = h(s+t') \qquad \text{(II-41)}$$

Ceci est illustré par la figure II-9. Cette propriété découle directement de la propriété d'accroissements indépendants et contribue largement à la confusion couramment commise entre taux défaillance et intensité de défaillance.

Nous présenterons dans le prochain paragraphe quelques modèles de croissance de fiabilité qui ont été développés ces vingt dernières années ; parmi ces modèles certains sont basés sur des NHPP.

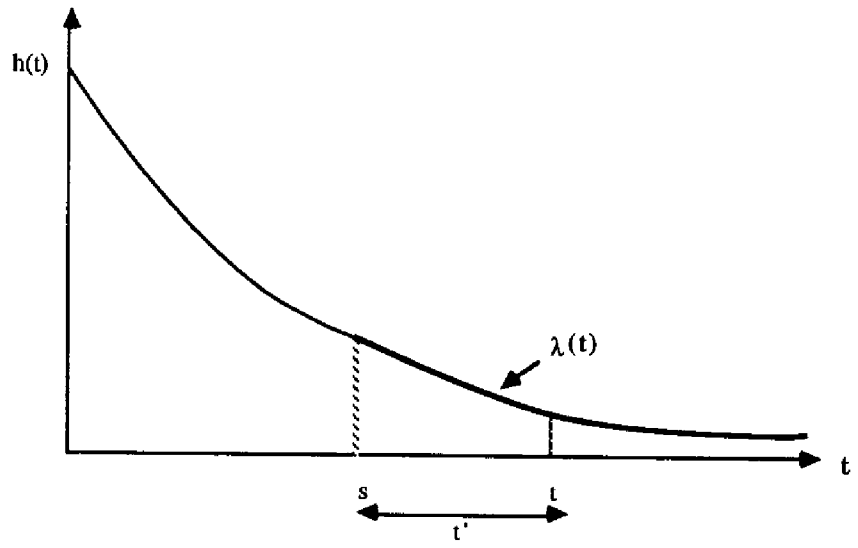


Figure II-9 : Taux λ(t) et intensité de défaillance h(t) pour un NHPP

II-2-2 Présentation de modèles de croissance de fiabilité

Nous avons vu que le processus de défaillance est caractérisé par la fonction intensité h(t), ou (ce qui est équivalent) par la fonction moyenne H(t) qui détermine l'allure de la croissance de fiabilité observée sur le logiciel. La présentation des modèles sera basée sur l'allure de h(t) indépendamment du cycle de vie du logiciel.

Le but n'est pas de détailler tous les modèles de croissance de fiabilité qui existent, mais de dégager des points communs, les différences essentielles entre ces modèles et surtout de montrer comment se situent ces modèles par rapport aux modèles de connaissance précédemment établis. Une revue très détaillée de ces modèles peut se trouver dans [Goe 83], Les articles [Dal 86, Mel 87] fournissent des exposés de l'état de l'art concernant la modélisation du logiciel et les modèles de croissance de fiabilité .

II-2-2-1 *Présentation générale*

Selon le point de vue adopté, on peut effectuer une classification des modèles de croissance de fiabilité afin de rendre compte d'un phénomène particulier. Il existe un certain nombre de classifications des modèles de croissance de fiabilité dans la littérature :

- la classification de [Ram 82] qui est basée sur les phases du cycle de vie du logiciel,
- la classification de [Goe 85] qui distingue les modèles à dénombrement des défaillances des modèles à taux de défaillance,
- la classification de [Mil 86] qui fait une répartition des modèles en deux catégories : modèles à taux déterministes et modèles doublement stochastiques,

- la classification que nous avons proposée dans [Kan 87a], [Sab 87] et [Lap 89] qui sépare les modèles à taux de défaillance des modèles à intensité de défaillance.

Une autre classification pourrait consister à distinguer les modèles qui tiennent compte explicitement des deux premières sources d'incertitude présentées dans le premier chapitre (sur les entrées et sur le programme) de ceux qui considèrent le processus dans sa globalité tels que les NHPP.

Nous adoptons la classification que nous avons proposée et qui distingue les deux catégories suivantes :

- modèles basés sur l'expression du taux de défaillance et de la relation entre les taux successifs,
- modèles basés sur l'expression de l'intensité de défaillance.

Cette distinction est purement schématique car tout modèle est à la fois à taux et à intensité de défaillance selon qu'il s'intéresse au comportement du logiciel entre deux défaillances ou au processus de succession des défaillances. Il y a cependant une différence essentielle entre ces deux catégories :

- la première catégorie de modèles suppose que l'amélioration de la fiabilité se fait de façon discontinue au moment de l'introduction d'une nouvelle édition et exprime directement cette discontinuité,
- les modèles de la deuxième catégorie intègrent cette discontinuité.

Un certain nombre de modèles appartenant respectivement à ces deux catégories seront présentés ci-après.

II-2-2-2 *Modèles à taux de défaillance*

Parmi les modèles à taux de défaillance, on peut citer celui de Jelinski-Moranda [Jel 72], celui de Shooman [Sho 73], celui de Musa [Mus 75], le modèle géométrique [Mor 75], le modèle de Littlewood-Verral [Lit 73], le modèle de Littlewood [Lit 81], le modèle de Keiller-Littlewood [Kei 83] et le modèle "proportionnel" proposé dans [Gau 89b],...
A l'exception du dernier modèle, tous ces modèles correspondent au cas de la *correction immédiate* avec une intensité de défaillance *strictement décroissante* tendant vers *zéro* quand tend t vers l'infini.

En fait, les trois premiers modèles sont équivalents et seront référencés Jelinski-Moranda.
Les quatre premiers modèles supposent que l'écart entre deux taux de défaillance est déterministe. Les trois modèles de Littlewood considèrent le processus de défaillance comme

un processus doublement stochastique : ils considèrent que le taux de défaillance est lui même une variable aléatoire ayant une certaine distribution : ils tiennent ainsi compte à la fois de l'incertitude sur le choix d'un point dans le domaine d'entrée et de l'incertitude sur la version du programme.

Le modèle de Jelinski-Moranda présente les caractéristiques suivantes :
- taux de défaillance strictement décroissant, la décroissance étant déterministe et constante : $\lambda_{i-1} - \lambda_i$ = constante = ϕ,
- λ_1 proportionnel au nombre de fautes N initialement dans le programme $\lambda_1 = N\phi$, (toutes les fautes ont la même contribution au niveau du taux de défaillance),
- r=N et $\lambda_r = 0$.
Le modèle peut être étendu pour tenir compte d'une éventuelle introduction de fautes durant les corrections.

Le modèle géométrique quant à lui a les propriétés suivantes :
- le taux de défaillance est strictement décroissant,
- le pas de décroissance est lui même décroissant, $\lambda_{i-1} - \lambda_i = D\,(1-k)\,k^{i-1}$, avec D et k des constantes et k<1,
- r est non connu a priori et est déterminé par : $\lambda_r \approx 0$ pour $i \rightarrow \infty$.

Dans le modèle proportionnel [Gau 89b], les taux de défaillance sont liés par la relation :

$$\lambda_{i+1} = C_i\,\lambda_i$$

où les C_i sont des variables aléatoires positives pouvant être supérieures à 1 afin de permettre la prise en compte d'introduction de nouvelles fautes.
Ce modèle coïncide avec le modèle géométrique pour C_i = une constante déterministe.

II-2-2-3 *Modèles à intensité de défaillance*

Ce sont essentiellement les modèles NHPP. Notre choix s'oriente vers cette catégorie : les raisons essentielles de ce choix résident d'une part dans le fait que les hypothèses de modélisation sont moins restrictives (surtout en ce qui concerne le lien entre les processus de défaillance et de correction) et d'autre part leur souplesse d'utilisation.
Ceci est confirmé par le nombre de modèles qui ont été développés : en effet c'est la catégorie la plus répandue. Trois classes de NHPP peuvent être considérées selon l'allure de l'évolution de la fiabilité :
- intensité croissante, puis décroissante tendant vers 0 quand $t \rightarrow \infty$, ceci se traduit par une courbe h(t) en forme de cloche,
- intensité décroissante tendant vers 0 quand $t \rightarrow \infty$,
- intensité décroissante tendant vers une limite non nulle quand $t \rightarrow \infty$.

Intensité en forme de cloche

Nous avons vu que ce cas peut avoir lieu durant le test du logiciel où :
- l'effort de test est variable,
- des fautes peuvent être introduites lors des corrections,
- les délais entre défaillances et corrections peuvent accentuer l'influence des fautes liées (ou dépendantes),

ou en vie opérationnelle :
- si le profil ou les conditions d'utilisation varient de façon significative,
- lors de l'introduction d'une nouvelle version.

Rappelons que : $h(t) = H'(t)$.
$h'(t)=0$ correspond à un point de changement de courbure (inflexion) pour H(t) qui a alors la **forme d'un S**. Parmi ces modèles on peut citer : "delayed S-shaped" [Yam 83], "inflexion S-shaped" [Ohb 82, Ohb 84] et "Logistic NHPP" [Fuk 86]. L'expression de l'intensité de défaillance relative à ces modèles est donnée ci-après :

Delayed S-Shaped : $h(t) = N \Phi^2 t \exp(-\Phi t)$, paramètres : N, Φ. (II-42)

Inflexion S-Shaped : $h(t) = \frac{N \Phi^2 \exp(-\Phi t)}{1+x \exp(-\Phi t)}$, paramètres : N, x, Φ. (II-43)

Logistic NHPP : $h(t) = \frac{N m \Phi}{[1+m \exp(-\Phi t)]^2}$, paramètres : N, m, Φ. (II-44)

Intensité décroissante tendant vers zéro

Ces modèles traduisent une croissance de fiabilité monotone. Le premier modèle [Cro 77] consiste à interpréter un modèle développé par Duane depuis les années 60 pour traduire la croissance de fiabilité du matériel [Dua 64] en tant que NHPP. Il a été suivi par un modèle qu'on pourrait aussi qualifier de l'interprétation NHPP du modèle de Jelinski-Moranda : modèle exponentiel [Goe 79]. D'autres modèles ont été établis depuis : le modèle exponentiel différé [Yam 83], le modèle logarithmique [Mus 84, Mus 87] (qui peut aussi être considéré comme l'interprétation NHPP du modèle géométrique), le modèle de Weibull [Fuk 86]. L'expression de l'intensité de défaillance de ces modèles est donnée ci-dessous :

Modèle de Duane : $h(t) = \lambda \beta t^{\beta-1}$, paramètres : λ, β. (II-45)

Modèle exponentiel : $h(t) = N \Phi \exp(-\Phi t)$, paramètres : N, Φ. (II-46)

Modèle exponentiel différé : $h(t) = N \Phi \exp[-\Phi(t-c)]$, paramètres : N, c, Φ. (II-47)

Modèle logarithmique : $h(t) = \frac{\lambda}{\lambda \theta t + 1}$, paramètres : λ et θ. (II-48)

Intensité décroissante tendant vers une constante non nulle

Il n'existe à notre connaissance que deux modèles qui tiennent compte d'un taux de défaillance non nul à l'infini :

- le premier est une extension du modèle de Duane obtenu en superposant à la fonction intensité (qui est décroissante) une constante correspondant au taux limite afin de tenir compte des fautes matérielles non corrigibles [Fin 79]. Cette idée n'a pas été reprise et, à notre connaissance, le modèle n'a pas été appliqué à des cas réels ; ce modèle ne sera pas considéré dans la suite car la même opération pourrait être faite sur tous les autres modèles [Per 89],
- le deuxième a pour expression une loi de Cox hyperexponentielle [Cox 68].

Ce dernier modèle (noté HE) sera détaillé au paragraphe II-2-3.

Nous résumons dans le tableau de la figure II-10 les caractéristiques des différents modèles. Pour les modèles à taux de défaillance, nous donnons l'expression du taux de défaillance ; pour les NHPP, c'est l'expression de l'intensité de défaillance qui est donnée.

II-2-2-4 *Conclusions*

Les modèles précédents ont deux caractéristiques communes (qui ne sont pas tout à fait indépendantes) :

- l'accent est mis sur l'évolution du nombre cumulé de défaillances beaucoup plus que sur celle du taux de défaillance,
- mis à part les deux derniers modèles présentés, les autres supposent qu'au bout d'un certain temps toutes les fautes ont été détectées et éliminées et que le taux de défaillance tend vers zéro.

En fait il faut replacer ces modèles dans leur contexte : ces modèles ont été établis pour suivre le développement (et plus particulièrement le test) du logiciel ; le fait que le taux de défaillance tende vers zéro n'implique pas forcément qu'aucune défaillance du logiciel ne peut avoir lieu en vie opérationnelle. Ceci pourrait être interprété de la manière suivante : le fait que le nombre résiduel de fautes est estimé nul signifie que les tests appliqués au programme ont atteint leur limite et qu'ils ne sont plus efficaces pour détecter le type de fautes pour lequel ils ont été développés. Ceci signifie que le taux de défaillance est nul pour ce profil d'utilisation particulier. S'il s'agit du test unitaire il faut passer aux unités suivantes, s'il s'agit du test d'intégration on peut passer à la validation pré-opérationnelle avant de livrer le programme.

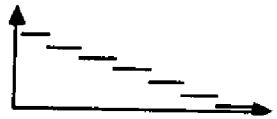
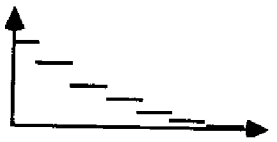
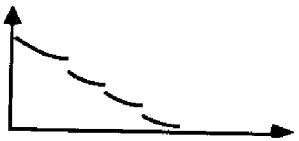
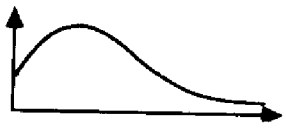
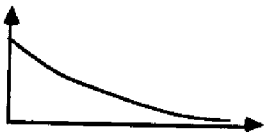
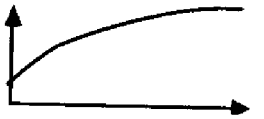
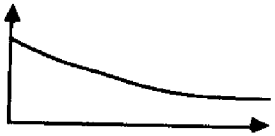
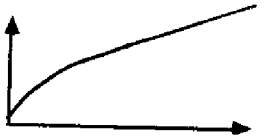
Type modèle	Modèle	Formulation	Paramètres	Allure
				λ(t) ou h(t) — H(t)
à taux de défaillance	Jelinski-Moranda	$\lambda_i(t) = (N-i+1)\,\phi$	N, ϕ	
	Géométrique	$\lambda_i(t) = D\,k^{i-1}$	D, k (k<1)	
	Littlewood-Verral	$\lambda_i(t) = \alpha / (t+\psi(i))$	α, $\psi(i) = \beta_1+\beta_2 i$	
à intensité de défaillance	S-shaped	$h(t) = N\,\phi^2\,t \exp -\phi t$	N, ϕ	 
	Exponentiel	$h(t) = N\,\phi \exp -\phi t$	N, ϕ	 
	Hyperexponentiel	$h(t)= \frac{\omega Z_1 e^{-Z_1 t} + \varpi Z_2 e^{-Z_2 t}}{\omega e^{-Z_1 t} + \varpi e^{-Z_2 t}}$	ω, Z_1, Z_2	 

Figure II-10 : Caractéristiques de quelques modèles de croissance de fiabilité

En développement, il est intéressant de suivre l'évolution du nombre de défaillances car il peut fournir des renseignements sur l'évolution du développement. Par contre en vie opérationnelle, un taux de défaillance est plus significatif, et c'est une des raisons pour lesquelles nous avons développé le modèle hyperexponentiel.

II-2-3 Présentation du modèle hyperexponentiel (HE)

En vie opérationnelle, l'utilisateur est intéressé par une estimation des mesures de la sûreté de fonctionnement de son système (fiabilité et disponibilité). La fiabilité est généralement exprimée en fonction du taux de défaillance du système. Nous avons donc besoin d'évaluer le taux de défaillance du logiciel au même titre que celui du matériel. Le modèle HE permet d'évaluer d'une part le taux de défaillance aussi bien en phase évolutive qu'en régime stabilisé (quand le taux de défaillance résiduel, λ_r, est atteint) et l'indisponibilité du système due aux défaillances du logiciel d'autre part.

Le modèle HE revêt donc deux aspects complémentaires selon qu'on s'intéresse à l'intensité de défaillance ou à la disponibilité. Ces deux aspects seront successivement étudiés dans les deux paragraphes suivants.

II-2-3-1 *Intensité de défaillance*

L'expression de l'intensité de défaillance a été choisie de telle sorte qu'elle puisse approcher au mieux les expressions très complexes de l'intensité de défaillance dérivées des modèles de connaissance afin de tenir compte des différentes politiques de maintenance déjà étudiées : croissance plus ou moins lente de la fiabilité.

L'intensité de défaillance est donnée par :

$$h(t) = \frac{\omega_1 Z_1 \exp(-Z_1 t) + \omega_2 Z_2 \exp(-Z_2 t)}{\omega_1 \exp(-Z_1 t) + \omega_2 \exp(-Z_2 t)} \qquad \text{(II-49)}$$

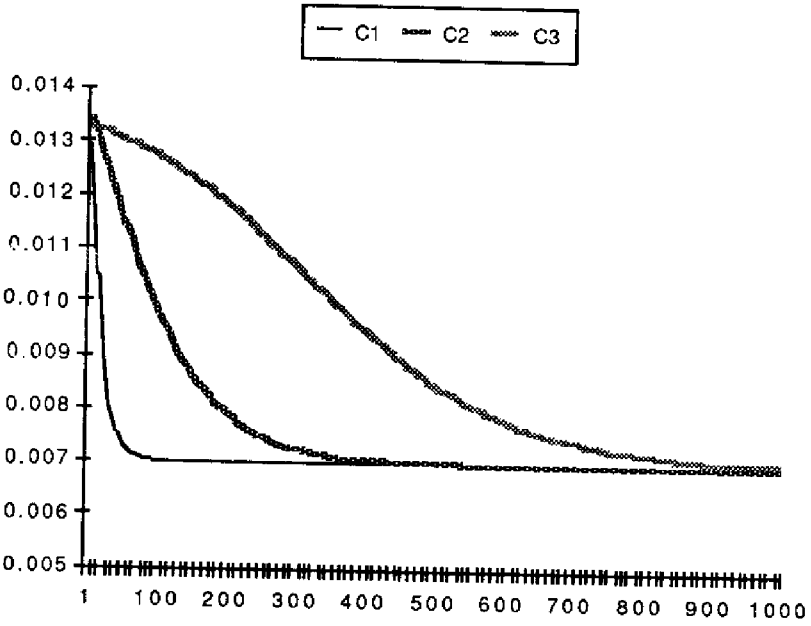
avec $0 \le \omega_1 \le 1$, $0 \le \omega_2 \le 1$ et $\omega_1 + \omega_2 = 1$.
Posons $\omega_1 = \omega$ et $\omega_2 = 1 - \omega = \varpi$.

h(t) est une fonction décroissante de h(0) à h(∞) donnés par :
$h(0) = \omega Z_1 + \varpi Z_2$
$h(\infty) = \inf(Z_1, Z_2) = \lambda_r$.

L'allure de la décroissance à l'origine est donnée par :
$h'(0) = -\omega\ \varpi\ (Z_1 - Z_2)^2$.

La décroissance est ajustable via l'assignation de valeurs aux paramètres (ω, Z_1, Z_2), ce qui permet de tenir compte de différentes politiques de maintenance.
En faisant varier ces paramètres on obtient des courbes qui coïncident avec celles de la figure II-4. La figure II-11 donne les courbes intensité de défaillance relatives au modèle HE en fonction de différentes valeurs des paramètres. Nous avons fixé $h(0) = \omega Z_1 + \varpi Z_2$ et $h(\infty) = Z_2$ afin de présenter, pour un même logiciel des vitesses d'évolution plus ou moins rapides :

- C1 illustre une croissance de fiabilité très rapide, et peut correspondre à un logiciel avec plusieurs instances et une politique de maintenance préventive (Cf. II-1-1-3-2),
- C2 correspond à une croissance de fiabilité progressive,
- C3 illustre une croissance de fiabilité assez lente, ce cas pouvant correspondre à une politique de correction par lot.



Courbe	ω	Z_1	Z_2
C1	0.1	$7\ 10^{-2}$	$7\ 10^{-3}$
C2	0.5	$2\ 10^{-2}$	$7\ 10^{-3}$
C3	0.9	$1.4\ 10^{-2}$	$7\ 10^{-3}$

Figure II-11 : Intensité de défaillance relative au modèle hyperexponentiel.

Posons $G(t) = \omega \exp(-Z_1 t) + \varpi \exp(-Z_2 t)$.

Les mesures de la sûreté de fonctionnement s'expriment facilement à l'aide de cette fonction :

$$h(t) = -\frac{G'(t)}{G(t)}$$

et :

$$H(t) = -\operatorname{Ln} G(t) \qquad \text{(II-50)}$$

Utilisant ces notations et les expressions (II-39) à (II-41) on a :

$$R(t' \mid s) = \frac{G(s+t')}{G(s)} \qquad \text{(II-51)}$$

$$f(t' \mid s) = -\frac{G'(s+t')}{G(s)} \qquad \text{(II-52)}$$

$$\lambda(t' \mid s) = -\frac{G'(s+t')}{G(s+t')} \qquad \text{(II-53)}$$

Le MTTF est alors :

$$MTTF = \frac{\frac{\omega_1}{Z_1}[\exp(-Z_1 s)] + \frac{\omega_2}{Z_2}[\exp(-Z_2 s)]}{\omega_1 \exp(-Z_1 s) + \omega_2 \exp(-Z_2 s)} \qquad \text{(II-54)}$$

Remarques

1) La généralisation de ce modèle serait d'introduire :

$$G(t) = \sum_{i=1}^{k} \omega_i \exp(-Z_i t) \qquad \text{avec : } \sum_{i=1}^{k} \omega_i = 1$$

afin d'approcher le plus possible les expressions de l'intensité de défaillance établies à l'aide des modèles d'action. Ces résultats ont été obtenus à l'aide d'une autre approche dans [Sol 89b]. Nous avons pris k=2, ceci est suffisant pour suivre le phénomène de croissance (Cf. figure II-11).

2) On aurait pu choisir d'autres formes analogues de h(t) traduisant aussi une croissance puis une stabilisation de la fiabilité. Le choix de l'expression spécifique de h(t) retenue permet d'avoir des expressions analytiques explicites de toutes les mesures de la sûreté de fonctionnement.

II-2-3-2 *Disponibilité*

Un modèle d'action compact permettant d'approcher les courbes d'indisponibilité, dans le cas où l'intensité de défaillance est décroissante et suit une loi de Cox hyperexponentielle, est donné par la figure II-12. Les paramètres (ω, Z_1, Z_2) correspondent à ceux déjà définis pour l'intensité de défaillance, avec $Z_2 < Z_1$. Dans les états 1 et 2 le logiciel délivre un service approprié et l'état D correspond au cas où le logiciel est défaillant (service inapproprié). Les probabilités initiales d'occupation des états sont respectivement : { ω ϖ 0 }. Le taux μ correspond au taux de correction du logiciel en régime stabilisé.

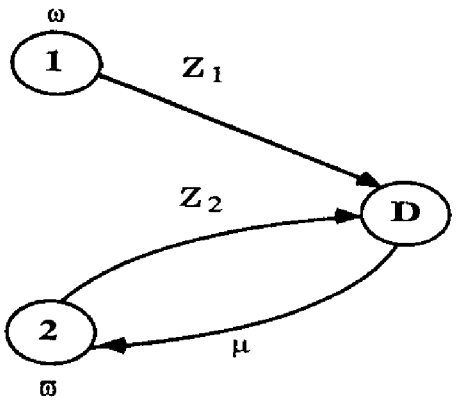


Figure II-12 : Modèle compact de disponibilité.

L'indisponibilité est évaluée à l'aide des méthodes classiques de résolution de graphes d'états markoviens ; en supposant (Z_1 , Z_2) << μ, il vient :

$$\bar{A}(t) = \frac{Z_2}{\mu} + \omega \frac{Z_1 - Z_2}{\mu} \exp(-Z_1 t) - \frac{\omega Z_1 + \varpi Z_2}{\mu} \exp(-\mu t) \qquad \text{(II-55)}$$

Pour $Z_1 = Z_2$ (Taux de défaillance constant), on retrouve l'expression classique de l'indisponibilité d'un système à taux de défaillance et de correction constants.

Le maximum de l'indisponibilité est obtenu pour t de l'ordre de quelques $\frac{1}{\mu}$ et est égal à :

$$\bar{A}_m = \frac{\omega Z_1 + \varpi Z_2}{\mu} \qquad \text{(II-56)}$$

La valeur asymptotique est donnée par :

$$\bar{A} = \frac{Z_2}{\mu} \qquad \text{(II-57)}$$

Cette valeur est atteinte au bout de quelques $\frac{1}{Z_1}$.

En utilisant l'intensité de défaillance, ces expressions peuvent être écrites sous la forme :

$$\bar{A}_m = \frac{h(0)}{\mu} \qquad \text{(II-56')}$$

$$\bar{A} = \frac{h(\infty)}{\mu} \qquad \text{(II-57')}$$

Ainsi l'indisponibilité maximale (resp. asymptotique) est égale au rapport de l'intensité de défaillance initiale (resp. résiduelle) au taux de correction du logiciel.

La figure II-13 donne les courbes d'indisponibilité pour les mêmes valeurs de (ω, Z_1, Z_2) que celles qui ont été prises pour tracer l'intensité de défaillance (figure II-11) et pour deux valeurs de μ. Ces courbes sont tout à fait comparables aux courbes d'indisponibilité associées au modèle de connaissance (figure II-7).

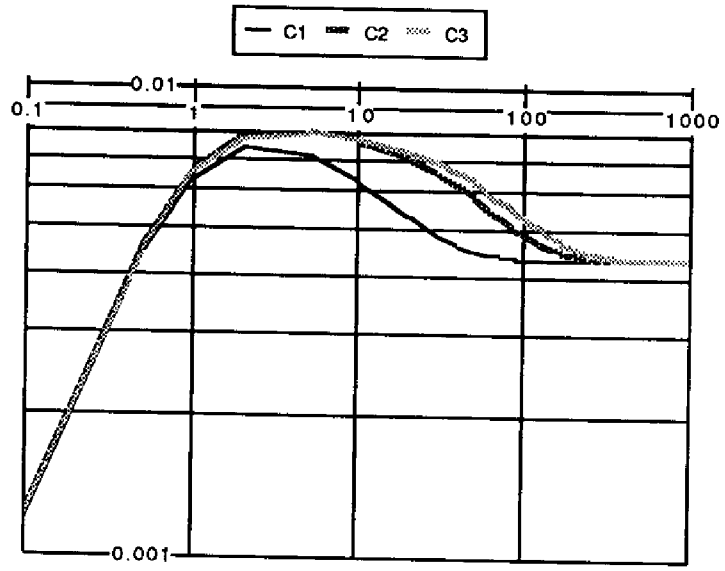
Ce modèle est très attractif car, bien que très simple, il permet de rendre compte du phénomène fondamental constitué par la croissance de fiabilité.

Remarques :

1) Le choix des valeurs des paramètres a été effectué afin d'obtenir des courbes d'indisponibilité comparables à celles obtenues à partir du modèle de connaissance. Dans [Kan 85] les paramètres ont été obtenus à l'aide d'un programme d'optimisation qui, à partir des différents λ_i estime (ω, Z_1, Z_2). Nous avons adopté la présente approche afin de montrer la cohérence entre l'intensité de défaillance et la disponibilité issues du même modèle considéré sous ses deux aspects.

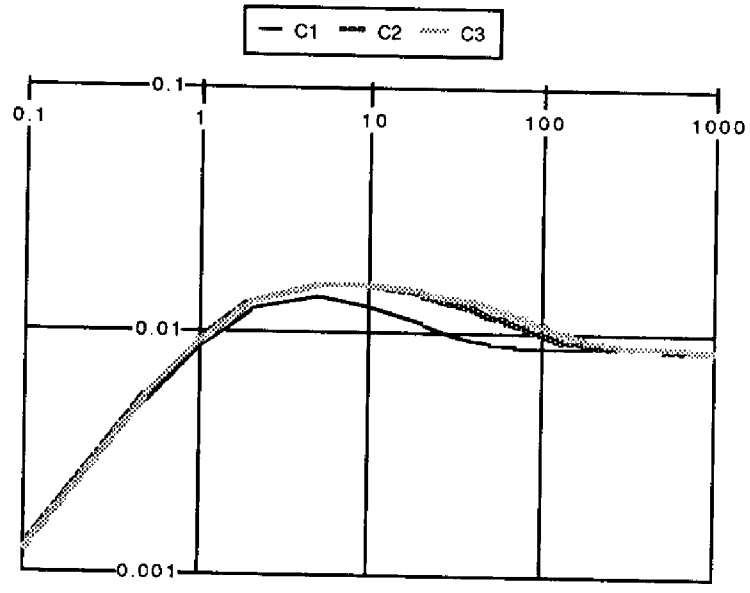
2) Le modèle étudié ci-dessus est légèrement différent de celui qui a été présenté dans [Lap 84a, b et Kan 85]. La différence réside dans les transitions de l'état D vers les états 1 et 2 : dans le modèle précédent il y avait une transition $\omega\mu$ vers 1 et une transition $\varpi\mu$ vers 2 ce qui impliquait une indisponibilité stationnaire égale à $\frac{Z_1 Z_2}{(\omega Z_2 + \varpi Z_1)\mu}$, le maximum d'indisponibilité ayant la même expression.

Les deux modèles donnent des courbes d'indisponibilité comparables avec toutefois des valeurs différentes de (ω, Z_1, Z_2). Notre préférence va vers celui qui est présenté dans ce mémoire car il permet de modéliser la croissance de fiabilité et de disponibilité avec la même interprétation des paramètres.



Courbe	ω	Z_1	Z_2	μ
C1	0.1	$7\ 10^{-2}$	$7\ 10^{-3}$	1.6
C2	0.5	$2\ 10^{-2}$	$7\ 10^{-3}$	1.6
C3	0.9	$1.4\ 10^{-2}$	$7\ 10^{-3}$	1.6

-a-



Courbe	ω	Z_1	Z_2	μ
C1	0.1	$7\ 10^{-2}$	$7\ 10^{-3}$	0.8
C2	0.5	$2\ 10^{-2}$	$7\ 10^{-3}$	0.8
C3	0.9	$1.4\ 10^{-2}$	$7\ 10^{-3}$	0.8

-b-

Figure II-13 : Courbes d'indisponibilité relatives au modèles HE

L'adoption de ce dernier modèle montre que dans le cas où l'on peut considérer que les taux de correction sont constants et ne dépendent pas du nombre de défaillances ayant eu lieu, l'étude de la croissance de disponibilité ne pose pas de problèmes particuliers et les résultats sont analogues à ceux obtenus à partir du modèle de fiabilité. Cette remarque nous amène à ne considérer que la croissance de fiabilité dans tout ce qui suit ; une autre raison de cet abandon provient du fait que les données concernant la disponibilité sont pratiquement inexistantes actuellement, la validation du modèle proposé à l'aide de données réelles ne peut malheureusement pas être entreprise.

II-2-4 Quelques commentaires sur les modèles

Parmi tous les modèles présentés (aussi bien de connaissance que d'action) aucun modèle ne tient compte explicitement de la quatrième source d'incertitude sur le comportement d'un logiciel (incertitude sur le profil d'utilisation) bien que l'importance de l'influence de la charge sur le processus de défaillance ne soit plus à démontrer [Iye 82]. Ceci pourrait être réalisé à l'aide de l'introduction de variables supplémentaires caractérisant l'environnement d'utilisation dans les différents modèles : les **variables explicatives** [Fon 85, McC 89]. Le principe de ces variables est très attrayant sur le plan théorique, cependant sa mise en œuvre est très difficile. En effet il faut pouvoir rendre compte de l'influence de l'environnement à l'aide de variables représentatives et simples afin de pouvoir les intégrer à la fois dans les modèles et dans la collecte des données. Le choix des "bonnes" variables explicatives n'est pas du tout évident. En fait l'idéal serait de pouvoir modéliser l'environnement et de combiner ce(s) modèle(s) aux modèles de croissance de fiabilité. Une tentative de modélisation du profil d'utilisation est effectuée dans [Ram 82] ; à notre connaissance les résultats de cette étude n'ont jamais été utilisés dans la pratique ; ceci est probablement dû à la difficulté de modéliser le profil d'utilisation de façon pratique.

On peut remarquer aussi qu'aucun modèle de croissance de fiabilité ne permet actuellement de modéliser une décroissance de fiabilité correspondant au début du développement suivie d'une croissance de fiabilité puis d'une fiabilité stationnaire différente de 1 (taux de défaillance résiduel non nul). Mais est-il nécessaire d'avoir un modèle de plus ? En pratique et en absence de ce type de modèle, l'étude de fiabilité sur plusieurs phases peut être effectuée grâce à l'utilisation soit de deux modèles différents (chacun étant plus adapté à la nature des données de la phase correspondante), soit du même modèle (avec des valeurs différentes des paramètres) en partitionnant l'ensemble des données en paliers. Ceci sera développé plus en détail dans les chapitres suivants.

II-2-5 Conclusions

Dans ce paragraphe nous avons présenté et analysé quelques modèles de croissance de fiabilité classiques que nous avons situés par rapport aux modèles de connaissance établis dans le paragraphe précédent et un modèle (le modèle hyperexponentiel) qui permet de modéliser à la fois la croissance de fiabilité et de disponibilité.

Une attention tout particulière a été portée aux modèles d'action basés sur les processus de Poisson non homogènes (NHPP). Ces modèles permettent de s'affranchir d'un certain nombre d'hypothèses de modélisation supplémentaires dont la plus contraignante est celle concernant le lien entre les processus de défaillance et de correction.

Concernant le modèle de disponibilité nous avons montré que les résultats obtenus à partir du modèle d'action proposé sont analogues à ceux obtenus à partir du modèle de fiabilité. Pour cette raison et par manque de données réelles pour le valider, ce modèle ne sera pas considéré dans ce qui suit.

II-3- CONCLUSIONS

Dans ce chapitre nous avons :
- établi différents modèles de connaissance caractérisant le comportement d'un logiciel en phases évolutive et stabilisée,
- présenté brièvement les principales caractéristiques de quelques modèles de croissance de fiabilité et leur lien avec les modèles de connaissance,
- présenté le modèle hyperexponentiel qui permet de modéliser à la fois la croissance de fiabilité et de disponibilité.

Les modèles de connaissance (relatifs à la fiabilité) établis correspondent à des politiques de correction différentes : correction après chaque défaillance (immédiate) et correction après un nombre moyen de défaillances.
Nous avons aussi considéré le cas d'un logiciel avec plusieurs instances sur des sites différents et nous avons étudié le cas où la fréquence des corrections sur un site moyen est inférieure à celle des défaillances et le cas où elle est supérieure à celle des défaillances. Ceci nous a permis de montrer l'influence bénéfique de la maintenance préventive sur l'évolution de la fiabilité du logiciel.
Cette modélisation a permis de bien montrer l'influence de différentes politiques de correction sur la décroissance de l'intensité de défaillance.
Nous avons ensuite montré comment ces modèles peuvent être généralisés au cas de changement de spécifications : maintenances adaptative et perfective.

Les expressions des mesures de la fiabilité étant très complexes, nous avons proposé d'approcher les modèles de connaissance par des modèles d'action. Notre choix est plus orienté vers les modèles basés sur la théorie des processus de Poisson non homogènes (NHPP) car ils ne supposent aucune relation entre les processus de défaillance et de correction.

Enfin, nous avons présenté quelques modèles de croissance de fiabilité existants en mettant l'accent sur les modèles basés sur les processus de Poisson non homogènes et plus particulièrement le modèle hyperexponentiel qui est le seul modèle permettant d'évaluer le taux de défaillance résiduel d'un logiciel en vie opérationnelle ainsi que la disponibilité.

La disponibilité ne sera pas considérée dans ce qui suit ; ceci est essentiellement dû d'une part au fait que les résultats obtenus par le modèle de disponibilité sont analogues à ceux obtenus à l'aide du modèle de fiabilité, et au manque de données réelles permettant de valider le modèle proposé d'autre part.

Selon la forme de l'intensité de défaillance (ou de la moyenne) certains modèles semblent plus appropriés que d'autres pour l'évaluation de la fiabilité d'un logiciel en développement ou en vie opérationnelle ; nous verrons dans le prochain chapitre comment choisir un modèle qui correspond à l'évolution réelle de la fiabilité d'un logiciel.

Dans tout ce chapitre le logiciel a été considéré comme une boîte noire, la prise en compte de la structure interne du logiciel sous forme de composants sera considérée dans le chapitre IV.

DEUXIEME PARTIE

APPROCHE GLOBALE POUR L'EVALUATION DE LA CROISSANCE DE FIABILITE

Les objectifs de cette seconde partie sont de définir et de mettre en œuvre une méthode globale permettant de quantifier la fiabilité d'un logiciel en phases évolutive et stabilisée. Cette méthode est constituée de deux étapes :

- une étape relative à l'étude du système concerné et des données collectées : fonctionnalités, architecture, environnement et profil d'utilisation, décomposition du logiciel, recensement des modes de défaillance, collecte de données, analyse qualitative de ces données et répartition par composant et par mode de défaillance,...
- une étape d'appréciation et d'évaluation de la fiabilité : prétraitement des données et application de modèles de croissance de fiabilité du logiciel.

Comme pour toute étude de fiabilité, l'étape relative au système est fondamentale car elle permet de bien comprendre les phénomènes mis en jeu et d'interpréter les résultats de la seconde. Elle est étroitement liée au système étudié et à son environnement. La collecte de données relatives aux défaillances du logiciel en constitue la majeure partie aussi bien en développement qu'en vie opérationnelle. Il n'est pas inutile d'insister sur l'importance de l'organisation de cette collecte et de son intégration au processus de développement. Un des objectifs de cette organisation doit être, en particulier, de ne pas augmenter considérablement la charge du personnel et de réduire de ce fait le coût de la collecte. La sensibilisation du personnel à la nécessité d'une telle collecte et aux bénéfices qu'elle peut engendrer aussi bien à court terme qu'à long terme constitue une condition nécessaire à sa mise en place et à sa qualité. Des méthodes de collecte de données valides sont proposées dans [Basi 84 et Kâa 89]. Des exemples d'analyses qualitatives de données (plus au moins complètes) peuvent être trouvés dans [Gla 81, Nag 82, Ros 82 ou Tro 86].

Le prétraitement des données consiste d'une part à s'assurer que les données recueillies constituent effectivement un ensemble homogène et d'autre part à déterminer l'allure de l'évolution de la fiabilité au cours du temps. La vérification de l'homogénéité des

données est réalisée grâce à la méthode de recherche des données douteuses (appelées également données aberrantes) et l'évolution de la fiabilité est donnée par les tests de tendance. L'évaluation de la fiabilité est réalisée grâce à l'utilisation des modèles de croissance de fiabilité présentés dans la première partie.

L'originalité et l'avantage de la méthode que nous proposons résident dans l'analyse des données et l'interprétation simultanée des résultats issus des tests de tendance et des données douteuses avant l'application des modèles de croissance de fiabilité. Le choix de la classe de modèles à appliquer est fonction des résultats obtenus à l'aide de ces tests, un modèle qui ne permet pas de tenir compte d'une éventuelle décroissance de fiabilité ne sera appliqué qu'en cas de croissance effective.

Nous avons déjà utilisé cette méthode pour l'évaluation de la fiabilité du logiciel de deux autocommutateurs téléphoniques [Kan 87c, Sab 87, Kan 88d], les principaux résultats relatifs à ces systèmes sont présentés dans le dernier chapitre de ce mémoire.
Le travail présenté dans ce mémoire permet d'approfondir et de discuter certains points plus particulièrement ; nous donnons une vue à la fois plus générale et plus fine des phénomènes étudiés. Nous introduisons les notions de tendance globale et locale, nous développons les tests correspondants et nous étendons le test de Laplace au cas où la variable aléatoire considérée est le nombre de défaillances par unité de temps.

Dans les chapitres constituant cette partie, nous ne détaillons pas la première étape de la méthode qui est très spécifique à l'application considérée et nous nous intéressons tout particulièrement à la seconde étape. Le chapitre III est consacré au prétraitement des données. Les aspects liés à l'application des modèles de croissance de fiabilité seront traités dans le chapitre IV. Le chapitre V présente les différentes étapes et les résultats de l'évaluation de la fiabilité du logiciel de deux autocommutateurs téléphoniques à l'aide de la méthode présentée dans les chapitres III et IV.

CHAPITRE III

PRETRAITEMENT DES DONNEES DE CROISSANCE DE LA FIABILITE

INTRODUCTION

Le prétraitement des données consiste d'une part à rechercher les données douteuses et à effectuer des tests de tendance afin de délimiter les zones de croissance et de décroissance de fiabilité. Ces deux points seront successivement étudiés. Nous donnerons ensuite quelques recommandations pratiques concernant l'utilisation de ces différents tests. Ces tests seront ensuite appliqués à des données prélevées sur des systèmes réels par le RADC [Mus 79].

III-1- RECHERCHE DES DONNEES DOUTEUSES

Afin de suivre l'évolution de la fiabilité d'un logiciel en développement ou en vie opérationnelle et d'évaluer ses mesures de sûreté de fonctionnement, un certain nombre de données sont prélevées. Les données recueillies doivent être à la fois fiables et aussi complètes que possible, car toute évaluation repose sur la qualité de ces données. Il arrive cependant que, malgré toutes les précautions prises, certaines données sont, soit entachées d'erreur, soit recueillies dans des conditions différentes des conditions "normales" d'utilisation. Ces données sont qualifiées "d'aberrantes" ou douteuses et doivent être identifiées et éventuellement écartées.

Le procédé le plus sûr pour vérifier que les données constituent bien un ensemble homogène consiste à examiner attentivement les conditions d'utilisation du système et de collecte des données. Dans la réalité ceci est difficilement réalisable. Il faut donc faire appel à des critères statistiques pour chercher les données douteuses.

Les données douteuses repérables sont à chercher en priorité parmi les valeurs extrêmes c'est-à-dire les valeurs minimale et maximale. En effet une donnée douteuse proche des valeurs moyennes prises par la variable aléatoire est difficilement détectable. La connaissance de l'effectif des données (n), de la moyenne μ et de l'écart type σ permet de déterminer des seuils

au delà des quels une observation peut être considérée comme douteuse. L'élimination d'une donnée douteuse faisant varier n, μ et σ et donc les seuils correspondants, l'identification de toutes les valeurs douteuses doit se faire étape par étape.

Méthode

Afin de détecter de telles valeurs, l'ensemble complet des intervalles entre défaillances observés classés dans l'ordre chronologique : $(t_1, t_2, \ldots t_n)$, est rangé par ordre croissant (appelé aussi ordre statistique) : $(t_{(1)}, t_{(2)}, \ldots t_{(n)})$.
Le but est de procéder à la vérification de l'une des hypothèses suivantes :

- l'ensemble des données constitue un échantillon homogène,
- l'observation de l'un des événements a été effectuée en présence de fortes variations dans les conditions, de telle sorte que cet événement doit être considéré comme douteux et éventuellement éliminé de toute étude ultérieure.

Cette vérification est réalisée à l'aide de l'étude de l'écart entre les intervalles inter-défaillances observés et la moyenne de ces intervalles, $v_i(n)$ [Aïv 70], définis par :

$$v_i(n) = \frac{|\mu - t_{(i)}|}{\sigma},$$

où μ et σ sont respectivement la moyenne et l'écart type empiriques définis par :

$$\mu = \frac{1}{n} \sum_{i=1}^{n} t_i \qquad \sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (\mu - t_i)^2. \qquad \text{(III-1)}$$

Comme on s'intéresse aux valeurs extrêmes seules $v_1(n)$ et $v_n(n)$ interviennent :

$$v_1(n) = \frac{\mu - t_{(1)}}{\sigma}, \qquad v_n(n) = \frac{t_{(n)} - \mu}{\sigma}. \qquad \text{(III-2)}$$

Il existe des tables [Aïv 70] qui déterminent la valeur maximale $v_{max}(n,\alpha)$, pour un seuil de signification α donné et une taille d'échantillons n donnée. Si l'une des deux inégalités :

$$v_1(n) < v_{max}(n,\alpha), \quad v_n(n) < v_{max}(n,\alpha) \qquad \text{(III-3)}$$

n'est pas remplie, l'observation correspondante est qualifiée de douteuse.

L'interprétation de α en tant que probabilité d'éliminer à tort l'observation douteuse n'est rigoureuse que si les t_i sont issus d'une population gaussienne ; dans les autres cas la méthode reste valable mais l'interprétation probabiliste de α est plus ou moins approchée.

La procédure que nous proposons permet d'identifier une seule valeur douteuse : la valeur la plus élevée ou la plus faible (éventuellement deux, si les deux inégalités ne sont pas vérifiées à la fois) ; afin d'identifier toutes les autres, il faut éliminer cette valeur et appliquer à nouveau la même procédure jusqu'à ce que les relations (III-3) soient vérifiées. Une telle procédure est facilement automatisable.

Application à la croissance de fiabilité

Bien que le test utilisé soit très rigoureux pour une population gaussienne, son utilisation en présence de variation de la fiabilité doit être effectuée avec certaines précautions ; c'est pour cela que la décision d'éliminer ou de garder ces valeurs est très subjective en cas de croissance de fiabilité et dépend des objectifs recherchés. En effet :
- une valeur très élevée peut indiquer une grande croissance de fiabilité due à l'amélioration de la qualité du logiciel, ou provenir du fait que l'enregistrement de cette donnée a été effectué avec du retard, ou simplement du fait que le système a été utilisé au "ralenti" pendant une certaine période,... ou qu'on a oublié d'enregistrer une défaillance.
- une suite de valeurs faibles peut correspondre à une phase faisant suite à des changements de spécifications ou à une accélération du développement.

Une valeur trop élevée ou trop faible, isolée et située en milieu de la période d'observation résulte vraisemblablement d'une "anomalie" et n'est pas forcément représentative du comportement du logiciel ; son élimination permettrait de ne garder que les données les plus vraisemblables et les plus significatives.

A notre avis, l'**élimination d'une donnée** ne doit pas être faite avant application des modèles. En particulier, en présence de croissance ou de décroissance de fiabilité, une fois ces données identifiées, il faut impérativement les garder si elles sont :

. très faibles et qu'elles se situent en début de la collecte,

. très élevées et qu'elles se situent en fin de période de collecte.

Dans tous les autres cas nous conseillons d'appliquer les modèles de croissance de fiabilité en gardant ces données, elles ne seront écartées que si les résultats obtenus s'écartent de façon significative de la réalité, écart dû essentiellement à l'influence de cette valeur.

III-2- TESTS DE TENDANCE

Avant d'appliquer des modèles de croissance de fiabilité à des données il faut s'assurer du sens de variation de la fiabilité. Ceci est réalisé à l'aide de tests de tendance qui permettent de voir si la fiabilité "s'améliore" ou se "détériore" au cours du temps. La tendance sera

étudiée soit sur les variables aléatoires intervalles entre défaillances soit sur le nombre cumulé de défaillances soit sur le nombre de défaillances par unité de temps. Les valeurs prises par ces variables résultent de la combinaison de plusieurs facteurs : domaine de défaillance du logiciel, profil d'utilisation, modifications effectuées. Une détérioration de la fiabilité n'est pas forcément synonyme de dégradation du logiciel, elle peut résulter d'une variation du profil d'utilisation. Si ces conditions sont homogènes, la décroissance de la fiabilité traduit effectivement une dégradation du logiciel. En fait ces variables permettent de suivre l'évolution de la fiabilité apparente c'est-à-dire telle qu'elle est perçue par l'utilisateur.

III-2-1 Tendance globale et tendance locale

Les tests de tendance sont effectués sur toute ou partie des données collectées. Considérant un ensemble de données, si le résultat du test à un instant donné est basé sur toutes les observations antérieures à t, la tendance indiquée par ce test est une **tendance globale** ; si au contraire ce résultat est uniquement fonction des observations qui ont eu lieu dans le voisinage de t, la tendance indiquée est une **tendance locale**. La tendance globale peut ainsi masquer le sens de variation locale de fiabilité.

Les notions de sens de variation locale et globale de la fiabilité sont fondamentales pour l'application des modèles de croissance de fiabilité et le choix des données à fournir aux modèles. Quand on utilise toutes les données pour l'application de modèles, c'est la tendance globale qui intervient, par contre si on effectue une partition des données en sous-ensembles, c'est la tendance locale qui détermine cette partition. Ce point sera détaillé dans le paragraphe III-3.

Indépendamment de la nature de la tendance qu'ils permettent d'identifier, deux catégories de tests de tendance peuvent être distinguées : des tests graphiques et des tests "analytiques" [Asc 84]. Les tests correspondant à la première catégorie constituent plutôt un moyen de vérification visuel de la croissance de fiabilité, ceux de la seconde catégorie sont plus rigoureux.
Ces deux catégories sont successivement présentées et discutées en fonction de leur faculté à détecter les tendances locale et globale.

III-2-2 Tests graphiques

Les tests graphiques permettent de visualiser par exemple :

- le nombre cumulé de défaillances sur la période d'observation, N(t),
- le nombre de défaillances sur des intervalles de temps courts (ces intervalles seront appelés unités de temps),

- la moyenne des intervalles de temps entre défaillances : test de la moyenne arithmétique.

Les deux premiers tests correspondent respectivement aux tracés des estimateurs de la moyenne et du ROCOF du processus : H(t) et h(t). Bien que ces deux grandeurs soient étroitement liées, il est plus facile et plus "fiable" d'utiliser la première : il suffit en effet de reporter sur le graphique le nombre cumulé de défaillances sur la période considérée :

- une croissance de fiabilité est matérialisée par une courbure négative,
- à l'inverse une décroissance de fiabilité est matérialisée par une courbure positive,
- une fiabilité stabilisée entraîne une fonction N(t) linéaire.

Un système ayant présenté une décroissance de fiabilité suivie d'une croissance de fiabilité est caractérisé par un changement de courbure : point d'inflexion. La figure III-1 donne des exemples d'allure d'évolution de N(t). Ce test permet de détecter la tendance globale.

Pour le deuxième test (évolution du nombre de défaillances par unité de temps), le choix de l'unité de temps est primordial : pour certains types de données, selon l'unité choisie, on peut faire apparaître une tendance ou la masquer. Afin d'illustrer ce phénomène nous donnons l'exemple d'un système ayant eu respectivement :

5 3 6 6 3 6 2 5 1 0

défaillances sur des unités de temps de même longueur ; au vu de ces chiffres, mise à part une croissance de fiabilité à la fin, il n'y a pas de tendance générale marquée.
Si nous prenons une unité de longueur double, nous aurons respectivement :

(3+5=8) 12 9 7 1

défaillances, ce qui indique une légère décroissance de fiabilité suivie d'une forte croissance. Néanmoins, les deux regroupements indiquent une croissance de fiabilité à la fin.
Ce test permet de détecter la tendance locale.

Le test de la moyenne arithmétique consiste à calculer la moyenne arithmétique ξ_k des intervalles entre défaillances observés (t_i) jusqu'à la k ième donnée : si les ξ_k forment une suite globalement croissante (décroissante) on en déduit une croissance (décroissance) de fiabilité.

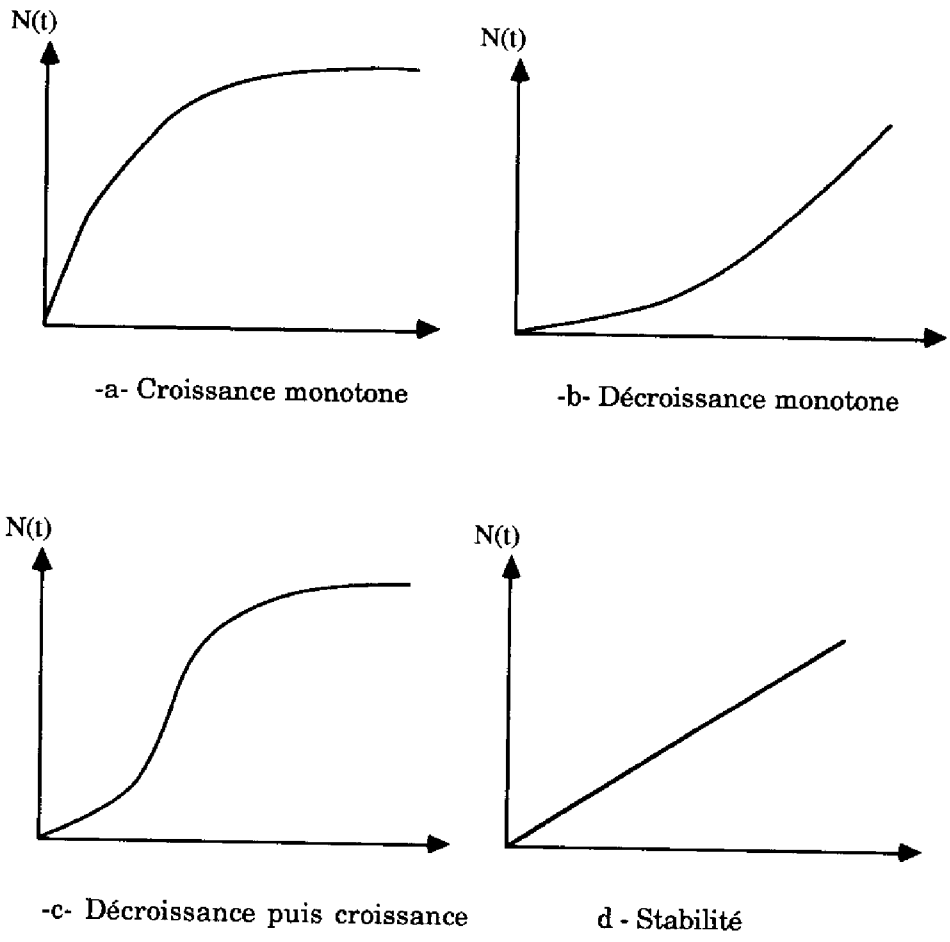


Figure III-1 : Variation de N(t) et tendance

La croissance est observée directement sur les valeurs successives des ξ_k. En fait, la croissance observée ne dénote pas la croissance effective des observations car, comme toute moyenne, elle est atténuée ; elle donne néanmoins une bonne indication sur le sens de variation de la fiabilité. Le choix de ce test est motivé par le fait qu'il est directement attaché à la grandeur physique perçue par l'utilisateur.

Le sens de variation de ξ_k donne la tendance globale car il tient compte de toutes les observations antérieures à k. La tendance globale peut ainsi masquer le sens de variation local de la fiabilité. Si on veut faire apparaître la tendance locale ce test doit être modifié comme suit : les données sont regroupées en sous-ensembles disjoints (appelés paquets) ; chaque paquet contient m données successives et la moyenne est évaluée par paquet. Le nombre m peut être fixe et le même pour toute la période considérée ou variable : on évalue la moyenne des intervalles entre défaillances par périodes de longueur fixe.

III-2-3 Tests analytiques

Parmi l'ensemble des tests présentés dans [Asc 84], nous avons sélectionné ceux correspondant aux tests de Laplace et de superadditivité. Le premier teste l'hypothèse d'un processus de Poisson homogène (HPP), l'alternative étant une **tendance monotone** alors que, pour le second, l'alternative est une **tendance non monotone**.
Une suite de variables aléatoires T_i, est dite à tendance monotone si elle est stochastiquement croissante ou décroissante, elle satisfait la condition :

$$F_{Ti}(t) < F_{Tj}(t) \quad \text{pour tout } i\geq 1,\ j>i \text{ et } t>0$$
ou
$$F_{Ti}(t) > F_{Tj}(t) \quad \text{pour tout } i\geq 1,\ j>i \text{ et } t>0 \qquad \text{(III-4)}$$

Dans le cas où la condition précédente n'est pas satisfaite complètement (pour tout i et tout j), on dira que la tendance n'est pas monotone.
Une étude comparative de tous les tests de tendance connus est effectuée dans [Gau 88].

III-2-3-1 *Test de superadditivité*

Dans le cas où la croissance de fiabilité n'est pas monotone pas à pas mais à long terme ou en moyenne[3], il est recommandé d'utiliser le test de superadditivité développé dans [Hol 74, Asc 78]. Ce test est défini comme suit :
H(x) étant la fonction moyenne du processus, H(x) est une fonction superadditive si l'inégalité suivante est vérifiée :

$$H(x_1) + H(x_2) \leq H(x_1 + x_2) \quad \text{avec} \quad 0 < x_1 + x_2 \leq s_n. \qquad \text{(III-5)}$$

Cette inégalité doit être stricte pour au moins un couple (x_1, x_2).

Une fonction superadditive correspond à une décroissance de fiabilité.
L'interprétation pratique de cette inégalité est la suivante : le nombre moyen estimé d'événements sur un intervalle de la forme [0,t] n'est pas plus grand que le nombre moyen estimé d'événements ayant eu lieu sur un intervalle de temps de même longueur mais commençant plus tard.

---

[3] Par analogie avec la terminologie employée dans [Bar 75, p. 84] concernant une distribution à taux de défaillance décroissant en moyenne (DFRA), une définition plus rigoureuse d'une intensité de défaillance décroissante en moyenne serait : h(t) est décroissante en moyenne si : $\frac{1}{t}\int_0^t h(x)\,dx$ est une fonction décroissante du temps, soit $\frac{H(t)}{t}$ fonction décroissante du temps.

Il est intéressant de remarquer que le test de superadditivité est équivalent à celui du ROCOF, et que de ce fait on rencontre les mêmes difficultés, mais énoncées de façon plus rigoureuse : choix des couples (x_1, x_2), ce qui revient au choix de la longueur des intervalles pour le ROCOF.
Selon le choix des instants x_1 et x_2, la tendance indiquée par ce test peut être locale ou globale.

III-2-3-2 *Test de Laplace*

Le test de Laplace a été développé pour une croissance de fiabilité monotone et est basé sur l'étude du signe de la différence entre la moyenne des instants d'occurrence des défaillances et le milieu de l'intervalle d'observation.
Soient :

- s_i, l'instant d'occurrence de la i ième défaillance, $s_i = \sum_{j=1}^{i} t_j$
- t_e, le temps total d'observation.

En supposant que le processus d'occurrence des défaillances est un processus de Poisson homogène, les instants s_i i=1,.., n des réalisations à l'intérieur de l'intervalle $[0, t_e]$ correspondent à l'ordre statistique d'une distribution uniforme sur $[0, t_e]$. Ainsi, le coefficient de tendance u (donné par (III-6)) tend vers une variable normale standardisée (l'approximation est significative à 5% pour n>3) :

$$u = \frac{c - m}{t_e} \sqrt{12\, n} \qquad \text{(III-6)}$$

avec :

n : le numéro de la dernière défaillance à l'intérieur de l'intervalle d'observation,
m : le milieu de l'intervalle d'observation, $m = t_e/2$,
c : le centre statistique (centroïde) qui correspond à la moyenne des instants d'occurrence des défaillances :

$$c = \frac{1}{n} \sum_{i=1}^{n} s_i.$$

Le signe de u correspond au signe de la différence entre le centre statistique (c) et le milieu de l'intervalle d'observation (m). L'interprétation de ce test est la suivante : en cas de croissance, les s_i auront tendance en moyenne à avoir lieu avant le milieu de l'intervalle d'observation.
La croissance (décroissance) de fiabilité est caractérisée par u négatif (positif).

Autres formulations du test de Laplace

1) Si le temps total d'observation t_e correspond à l'instant d'occurrence de la n ième défaillance (s_n), la dernière défaillance ne doit pas être prise en compte et ce test doit être modifié de la façon suivante :

$$u = \frac{c - m}{s_n} \sqrt{12\,(n-1)} \qquad \text{(III-6')}$$

avec : $m = \frac{s_n}{2}$ et $c = \frac{1}{n-1} \sum_{i=1}^{n-1} s_i$.

2) Le test de Laplace, tel qu'il a été présenté, est effectué sur l'ensemble des données alors que les tests graphiques sont faits en prenant en compte progressivement les données. Afin d'avoir une vision plus fine de l'évolution de la fiabilité, nous recommandons de l'appliquer progressivement sur un ensemble de plus en plus grand, voire pas à pas. Dans ce cas on évalue un ensemble de coefficients de tendance u(j) définis comme suit :

$$u(j) = \frac{c - m}{s_j} \sqrt{12\,(j-1)}, \qquad j=2,..n \qquad \text{(III-7)}$$

avec : $m = \frac{s_j}{2}$ et $c = \frac{1}{j-1} \sum_{i=1}^{j-1} S_i$.

3) Si on travaille sur les variables aléatoires nombre de défaillances par unité de temps, le test de Laplace doit être adapté à ces variables aléatoires de la façon suivante :

$$u(k) = \frac{c - m}{\sqrt{\frac{k^2-1}{12\,y_k}}} \qquad k=2,\ldots,p \qquad \text{(III-8)}$$

avec : $c = \frac{\sum_{i=1}^{k} (i-1)\, n_i}{y_k}$

$m = \frac{k-1}{2}$

p : nombre d'unités de temps sur l'intervalle d'observation,

n_i : nombre de défaillances durant l'unité de temps numéro i,

y_k : nombre cumulé de défaillances jusqu'à l'unité de temps k.

Le développement de l'établissement de u est donné à l'annexe 2.

Ainsi défini, le coefficient de tendance est évalué pas à pas c'est-à-dire au niveau de chaque unité de temps.

Nature de la tendance :

Quelle que soit la forme utilisée, le test de Laplace indique la tendance globale puisqu'il tient compte de toutes les données antérieures à l'indice considéré. La tendance locale peut être détectée grâce au sens de variation du coefficient de tendance :

- si u est positif, il y a décroissance globale de fiabilité :
    - dans le cas où u a tendance à croître la décroissance de fiabilité s'accentue,
    - si u a tendance à décroître la décroissance globale de fiabilité est atténuée par une croissance de fiabilité locale,
- si u est négatif, il y a croissance globale de fiabilité,
    - dans le cas où u a tendance à décroître la croissance de fiabilité est accentuée,
    - si u a tendance à croître, la croissance de fiabilité globale est atténuée par une décroissance de fiabilité locale.

Ceci est illustré par la figure III-2 où tous les cas de figure sont représentés ; le ROCOF associé à u(k) est donné afin d'indiquer la variation locale de fiabilité.

Les points de changement de tendance de u(k) sont les points d'inflexion de la courbe nombre cumulé de défaillances observées, N(t) (et on peut espérer qu'ils le soient aussi pour H(t) qui est son estimation par le modèle).

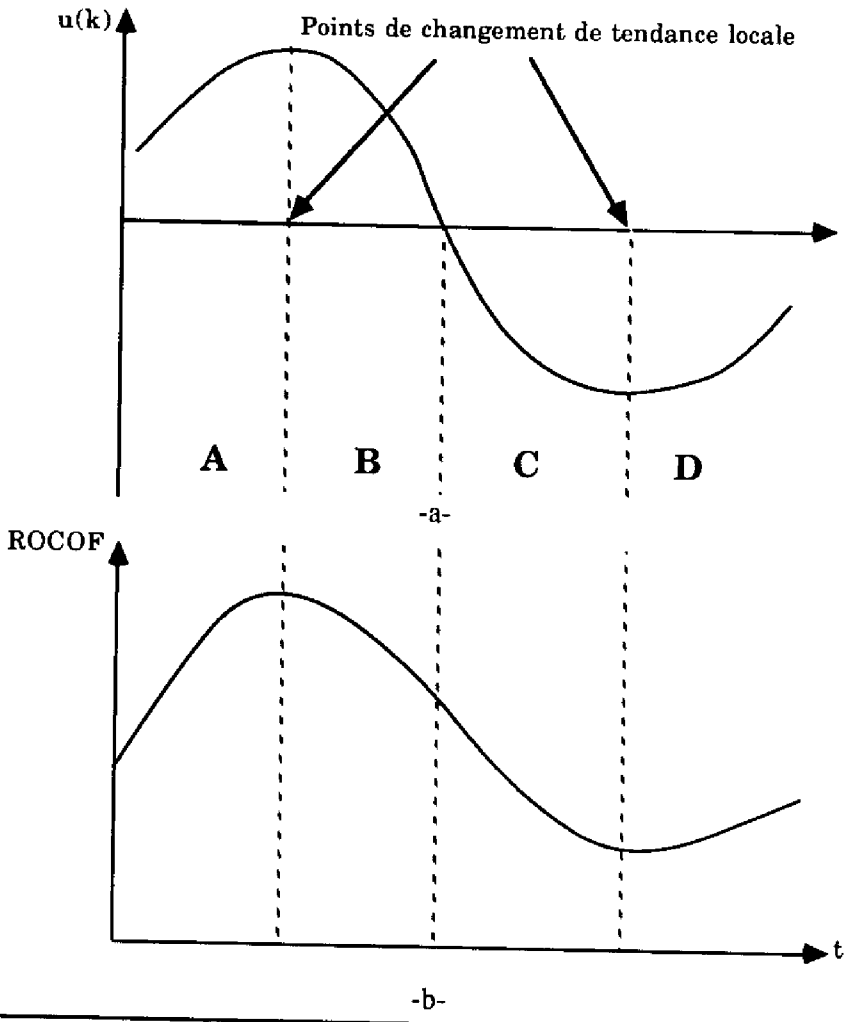
On peut remarquer qu'il existe deux types de point d'inflexion :

- type 1 : le coefficient est positif et change de tendance (transition A-B), cette situation est souhaitable car elle traduit une tendance à la croissance,
- type 2 : le coefficient est négatif et a tendance à croître (transition C-D), ceci dénote une certaine régression, il est conseillé de chercher une explication à ce phénomène.

Si on écarte les données relatives à la zone A de la figure III-2a et qu'on évalue le coefficient de tendance sur les zones B, C et D uniquement, ce coefficient devient négatif sur les trois zones puisque la tendance locale de B devient la tendance globale de B aussi. Après suppression des données de A, u reste négatif sur C et D mais son sens de variation ainsi que ses valeurs peuvent être modifiés selon les valeurs respectives des données appartenant aux différentes zones.

Remarque :

Nous n'avons considéré le test de Laplace que du point de vue nature de la tendance qu'il permet de détecter, l'étude des propriétés d'optimalité et de sa puissance dans le domaine de



-b-

u(k)	positif	négatif
décroissant	DF globale, CF locale (B)	CF globale et locale (C)
croissant	DF globale et locale (A)	CF globale, DF locale (D)

CF: Croissance de Fiabilité DF : Décroissance de Fiabilité

Figure III-2 : Coefficient de Laplace et ROCOF en fonction de l'évolution de la fiabilité

la croissance de fiabilité est effectuée dans [Gau 89b] et montre que ce test a d'excellentes propriétés d'optimalité dans de nombreux modèles, et qu'il est généralement pas possible de calculer sa puissance sur une alternative autre que le processus de Poisson homogène.

III-2-4 Lien entre les différents tests de tendance

Nous avons présenté un certain nombre de tests de croissance de fiabilité : des tests graphiques et des tests analytiques, les variables aléatoires étant soit les intervalles entre

défaillances soit le nombre de défaillances par unité de temps (ou cumulé sur tout l'intervalle d'observation).
Le choix de la variable aléatoire dépend des objectifs de l'étude de fiabilité (Cf. Chapitre IV): suivi du développement, planification de l'effort de test et de maintenance (dans ce cas on considère le nombre de défaillances) ou évaluation de la fiabilité.

La distinction entre test analytique et graphique n'est pas fondamentale (le test de Laplace tel qu'il est utilisé dans ce contexte peut aussi être considéré comme un test graphique), par contre il est important de connaître la nature de la tendance qu'il permet de mettre en évidence : tendance locale ou globale. La nature de la tendance relative aux différents tests est résumée dans le tableau de la figure III-3.

Test de tendance	Tendance globale	Tendance locale
Nombre cumulé de défaillances	x	
Nombre de défaillances / unité de temps		x
Moyenne arithmétique	x	
Moyenne arithmétique par paquet		x
Superadditivité	x	x
Laplace (signe de u)	x	
Laplace + sens de variation de u	x	x

Figure III-3 : Lien entre tests de tendance et la nature de la tendance.

La figure III-4 permet de mettre en évidence le lien qui existe entre le sens de variation du nombre cumulé de défaillances (N(t)), de l'intensité de défaillance et du coefficient de tendance de Laplace pour quatre cas type : croissance de fiabilité monotone, décroissance monotone, décroissance puis croissance et fiabilité stabilisée. Une fiabilité stabilisée est caractérisée par u=0, mais dans la pratique u peut osciller autour de zéro en prenant alternativement des valeurs légèrement négatives ou légèrement positives : $-2<u<+2$ indique une fiabilité quasiment stabilisée.

Dans la pratique les courbes représentant u(k) ne sont pas aussi lissées que celles indiquées sur la figure III-4 : afin de masquer les fluctuations dues à la dispersion des réalisations de la variable aléatoire, il est intéressant de travailler sur des données groupées (dans le cas où les données sont en nombre suffisant et où l'on s'intéresse au nombre de défaillances).

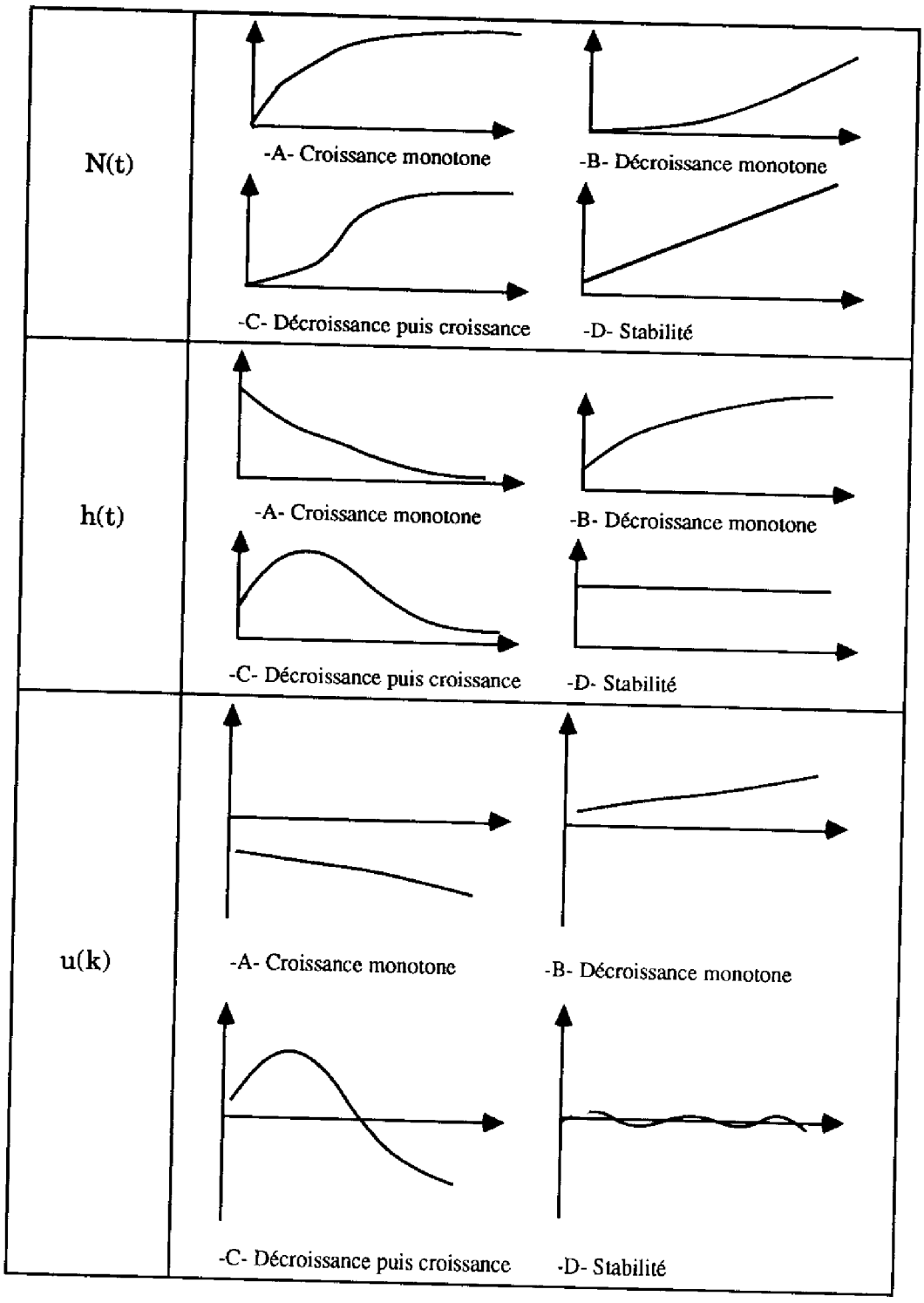


Figure III-4 : Sens de variation de N(t), h(t) et u(k)

III-3- RECOMMANDATIONS PRATIQUES

Disposant d'un ensemble de données de fiabilité relatives à un logiciel, nous pouvons d'une part détecter l'existence de données douteuses et, d'autre part, tester ou plutôt apprécier la tendance. C'est l'ensemble de ces résultats combiné à l'application des modèles de croissance de fiabilité, qui va permettre de prendre une décision en ce qui concerne l'élimination des données douteuses. Dans le cas où il y a croissance de fiabilité, les données douteuses faibles situées en début de période et élevées situées en fin de période doivent être gardées.

Les données douteuses isolées situées en milieu de période résultent généralement d'un comportement "anormal" dont il est souhaitable d'identifier l'origine ; ces données ne sont pas forcément représentatives du comportement du logiciel. Pour ces données, on appliquera les modèles de croissance de fiabilité en tenant compte de ces valeurs : si ces données sont gênantes pour obtenir des résultats, on les élimine et on applique à nouveau les modèles, ce qui permet d'estimer leur influence sur l'évaluation de la fiabilité du système. Ce n'est qu'après toutes ces opérations qu'il faut prendre la décision de les garder ou de les éliminer.

Du point de vue théorique les deux tests (test des données douteuses et test de tendance) sont indépendants, cependant l'élimination d'une donnée douteuse peut modifier le résultat des tests de tendance. Dans le cas où il y a un doute soit sur l'influence soit sur l'élimination de certaines données douteuses, on peut effectuer les deux tests indépendamment, écarter ces données douteuses, et refaire, le cas échéant, un test de tendance.

En ce qui concerne les tests de tendance, les tests graphiques constituent un bon indicateur visuel et rapide pour voir le sens de variation de la fiabilité. Les tests analytiques donnent des résultats plus quantitatifs. Le test de superadditivité semble plus adapté, car souvent la croissance de fiabilité n'est pas monotone pas à pas mais à long terme comme spécifié par ce test, néanmoins sa mise en œuvre est plus difficile que les tests précédents.

Une discussion comparative de la qualité de ces deux tests statistiques et de leurs avantages et inconvénients se trouve dans [Asc 78]. Pour notre part, par expérience, nous avons constaté que, sur le plan pratique, le test de Laplace est toujours largement suffisant : il faut cependant préciser qu'il est nécessaire de se servir à la fois du signe de u et de son sens de variation ; par expérience nous avons remarqué que dans le cas où u oscille entre -2 et +2 la fiabilité peut être considérée comme stabilisée.
Ce test sera effectué de façon systématique.

Dans le cadre de la méthode proposée nous préconisons d'utiliser de façon générale des représentations graphiques du nombre cumulé de défaillances, de la moyenne arithmétique et du test de Laplace ; le ROCOF sera utilisé dans le cas où ses résultats sont incontestables.

Les résultats de ces tests sont intéressants à la fois en soi et pour l'étape d'évaluation :

- L'étude de l'évolution de N(t), nombre cumulé de défaillances, constitue un bon indicateur de l'avancement du développement :
  - une courbure positive de N(t) en début de test est tout à fait admissible mais elle est alarmante si elle dure longtemps,
  - le changement de courbure indique le début de la période d'amélioration selon le profil d'utilisation en cours,
  - la stabilisation de N(t) indique que c'est le moment de passer à la phase suivante.

- Les mêmes résultats peuvent être obtenus en suivant l'évolution du ROCOF (le taux d'occurrence des défaillances), mais comme nous l'avons déjà signalé, les résultats peuvent dépendre du choix de l'unité de temps.

- Pour l'évaluation de la fiabilité ou du nombre de défaillances, on partitionne l'ensemble des données selon les tests de tendance de la façon suivante :
  - un modèle en forme de S sera appliqué sur un intervalle de temps durant lequel on a observé une décroissance de fiabilité suivi d'une croissance de fiabilité,
  - un modèle à croissance de fiabilité uniquement sera appliqué à un intervalle durant lequel on a observé une croissance de fiabilité.

Ces intervalles seront appelés **paliers** dans ce qui suit. La répartition en paliers est guidée par les points d'inflexion :

- un modèle en forme de S peut être appliqué entre deux points d'inflexion du type 2 (début de décroissance locale de fiabilité) et inclut forcément un point d'inflexion de type 1 (début de croissance locale de fiabilité) ,
- un modèle à croissance de fiabilité uniquement peut être appliqué entre un point d'inflexion type 1 et un point du type 2, les portions de la courbe comprises entre un point d'inflexion du type 2 et un point du type 1 (décroissance de fiabilité) ne seront théoriquement pas prises en compte pour l'évaluation par ce type de modèle ; dans la pratique ceci est pénalisant surtout quand le nombre de données est faible, généralement de telles données sont prises en compte quand même ce qui diminue considérablement la qualité de l'estimation.

III-4- APPLICATION AUX DONNEES RADC

Un certain nombre de logiciels de nature différente ont été observés sur des périodes plus ou moins longues. Les données correspondantes ont été publiées par le RADC dans [Mus 79]. Il faut préciser que les données disponibles sont rares dans ce domaine, en effet même quand elles existent les entreprises hésitent avant de les publier. Notre choix a porté sur ces données car elles constituent (malgré le manque de certaines informations relatives aux systèmes correspondants) un ensemble varié, permettant ainsi de traiter plusieurs cas afin d'interpréter les résultats de façon comparative.

Bien que cet ensemble de données existe depuis une dizaine d'années et qu'il ait souvent servi à valider et à comparer plusieurs modèles de croissance de fiabilité dans nombre de publications [Moa 81, Kei 83, Abd 86], à notre connaissance, aucun test de tendance n'a été appliqué à ces données (plus exactement publié) si ce n'est dans le cadre des travaux que nous avons menés [Kan 86 et Kan 88b] et dernièrement dans [Gau 88].

Les caractéristiques des logiciels étudiés sont résumées dans la figure III-5.

Les systèmes 1 à 6 et 8, 9 10 étaient encore en test alors que les autres systèmes étaient en vie opérationnelle. Il est important de noter qu'en vie opérationnelle aucune maintenance corrective n'a lieu : après défaillance, le système est relancé avec des données d'entrée différentes de celles qui ont entraîné la défaillance. Cependant, pour ces mêmes systèmes, à l'exception du système 13, il y a eu de légers changements de spécifications : ce sont des logiciels qui n'ont subi que de la maintenance adaptative ou perfective.

Les systèmes 11 A, B, et C correspondent à trois instances d'un même logiciel utilisées dans des environnements différents.

A l'exception du système 6 (pour lequel les intervalles de temps donnés sont des temps d'exécution), les temps sont des temps calendaires ; pour tous l'unité est la seconde.

Les données concernant les corrections ne sont pas aussi précises que celles correspondant aux défaillances : elles sont données en jours alors que celles des défaillances sont en secondes, nous ne pouvons donc les exploiter. En ce qui concerne la relation entre défaillances et corrections, nous n'avons pas réussi à dégager de règle générale : les fautes étaient corrigées au fur et à mesure de leur identification.

Numéro du système*	Nombre instructions	Nombre de programmeurs	Nombre de défaillances	Nature du système
1	21 700	9	136	C.C.T.R.
2	27 700	5	54	C.C.T.R.
3	23 400	6	38	C.C.T.R.
4	33 500	7	54	C.C.T.R.
5	2 445 000	275	831	Commercial T.R.
6	5 700	8	73	Commercial
7 (14C)	*****	110	36	T.R.
8 (17)	61 900	8	38	militaire
9 (27)	126 100	8	41	militaire
10 (40)	180 000	8	101	militaire
11A (SS1A)	*****	inconnu	112	Système d'exploitation
11B (SS1B)	*****	inconnu	375	Système d'exploitation
11C (SS1C)	*****	inconnu	277	Système d'exploitation
12 (SS2)	*****	inconnu	192	Système temps partagé
13 (SS3)	*****	inconnu	278	Traitement de texte
14 (SS4)	*****	inconnu	196	Système d'exploitation

T.R. : temps réel

C.C. : contrôle-commande

***** : plusieurs centaines de milliers (non précisé)

* numéro entre parenthèses : numéro d'identification dans Mus [79]

Figure III-5 : Présentation des systèmes étudiés

III-4-1 Recherche des données douteuses

Nous avons appliqué le test des données douteuses à chacun de ces systèmes. A part le système 11A, tous les systèmes possèdent des valeurs douteuses (au niveau de signification 5%). Pour chaque système la première valeur a été éliminée et on a refait passer le test, il s'est avéré que certains possèdent plusieurs valeurs douteuses. Les résultats sont résumés dans le tableau de la figure III-6. En fonction des résultats des deux tests (valeurs douteuses et tendance) nous avons sélectionné un sous-ensemble de systèmes significatifs et différents qui seront analysés plus en détail après l'application des tests de tendance. Nous pouvons néanmoins préciser d'ores et déjà que la majorité des données douteuses se situent en fin de période de collecte, ce qui pourrait être interprété comme une croissance de fiabilité.

Numéro du système	Nombre de données	$v_1(n)$	$v_n(n)$	$v_{max}(n)$	Nombre de données douteuses
1	136	0.63	5.31	3.5	8
2	54	0.67	4.58	3	2
3	38	0.63	3.52	2.88	4
4	54	0.31	5.54	3	6
6	73	0.48	5.74	3.2	11
7	36	0.71	3.23	2.86	5
8	38	0.45	5.59	2.88	4
9	41	0.49	5.15	2.91	4
10	101	0.39	5.59	3.4	9
11A	112	0.87	3.34	3.4	aucune
11B	375	0.70	5.47	4	12
11C	277	0.64	5.57	3.8	17
12	192	0.72	4.32	3.6	2
13	278	0.66	6.17	3.8	5
14	196	0.75	4.94	3.6	6

Niveau de signification 5 %

Figure III-6 : Recherche des données douteuses pour l'ensemble des systèmes

III-4-2 Tests de tendance

Les résultats de l'application du test de Laplace sur l'ensemble des données sont résumés dans le tableau de la figure III-7 : mis à part le système 12, tous les systèmes semblent dénoter une croissance de fiabilité globale en fin de période d'observation. Ceci doit être nuancé par la fait que d'un point de vue pratique, un coefficient de tendance compris entre -2 et +2 peut à la limite correspondre à une fiabilité stabilisée, ce qui est pratiquement le cas pour les systèmes 7, 11A, 11B, 12 et 14. Ce phénomène est beaucoup plus visible quand on applique le test de Laplace pas à pas ou le test de la moyenne arithmétique.

Les résultats de l'application du test pas à pas aux systèmes 2, 4, 9 et 14 sont donnés au paragraphe suivant.

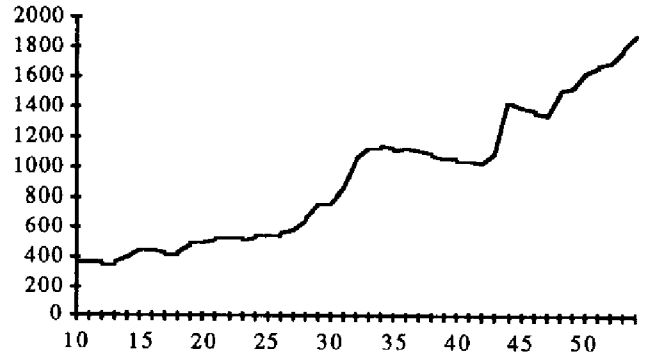
Numéro du système	Coefficient de Laplace	
1	- 9.10	
2	- 5.73	
3	- 6.13	
4	- 8.59	
6	- 3.64	
7	- 2.14	*
8	- 4.65	
9	- 5.16	
10	- 9.60	
11A	- 1.36	*
11B	- 0.73	*
11C	- 5.15	
12	+ 0.74	*
13	- 5.64	
14	- 1.78	*

Figure III-7 : Test de Laplace pour l'ensemble des systèmes

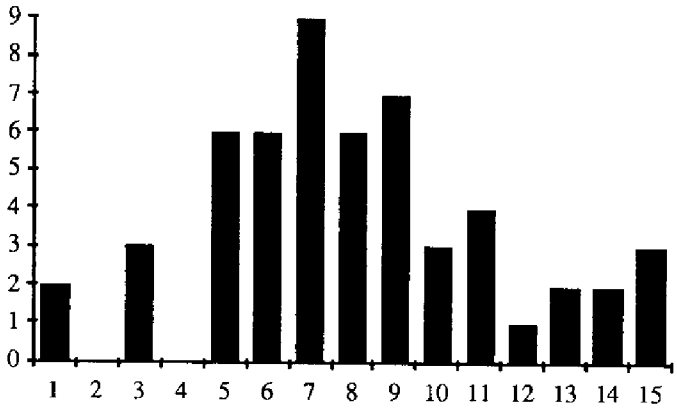
III-4-3 Analyse détaillée de quelques exemples

Pour le système 2 (dont le nombre total de défaillances observées est 54 et qui a un coefficient de Laplace global de -5.73) la moyenne des intervalles entre défaillances passe de 400 environ à 1800 (figure III-8a), avec une légère décroissance pour les données 35 à 42 ; cette décroissance pourrait éventuellement résulter d'une accélération du développement avant livraison. Deux données ont été détectées douteuses correspondant respectivement à la 43 ième et 47 ième donnée. Le tracé du nombre de défaillances par intervalles de 5 jours montre que le ROCOF est en forme de cloche, le maximum se situant vers les données 26-27 (figure III-8b). La courbe donnant le nombre cumulé de défaillances est bien en forme de S (figure III-8c). L'application des modèles de croissance de fiabilité sur l'ensemble des données et par fenêtre est donnée dans [Kan 87a].

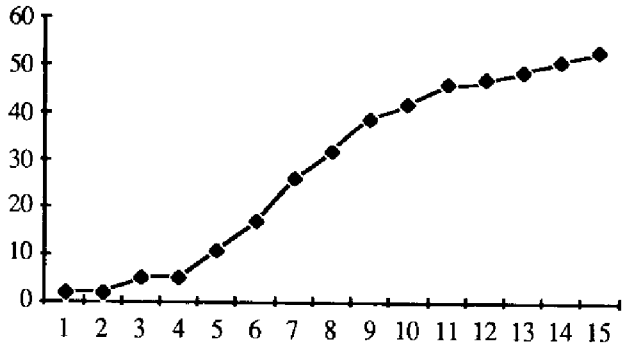
Le système 4 possède six données douteuses (pour un total de 54) qui ont respectivement pour numéros : 52, 51, 54, 36, 38 et 53 (l'ordre correspond à celui de leur identification en tant que données douteuses par applications successives du test). Pour ce système (figure III-9), on peut distinguer 3 ou 4 phases selon qu'on élimine ou non les quatre dernières données (qui ont d'ailleurs été qualifiées de douteuses). Cette forte croissance de fiabilité a peut être été à l'origine de la prise de décision d'arrêter le test : en effet tout de suite après, le système a été mis en opération.



a- Moyenne arithmétique en fonction du numéro de défaillance



b- Nombre de défaillances par période de 5 jours



c- Nombre cumulé de défaillances par période de 5 jours

Figure III-8 : Tests de tendance pour les données du système 2

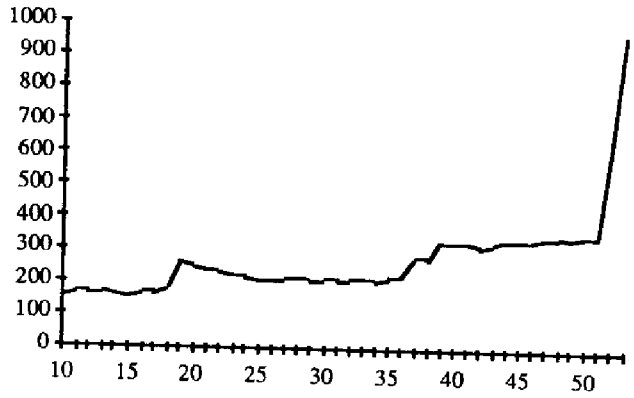
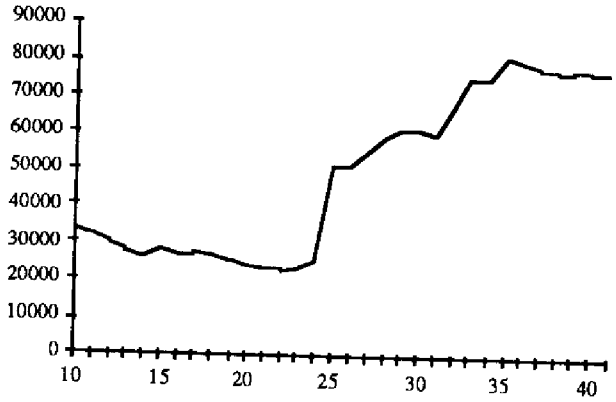
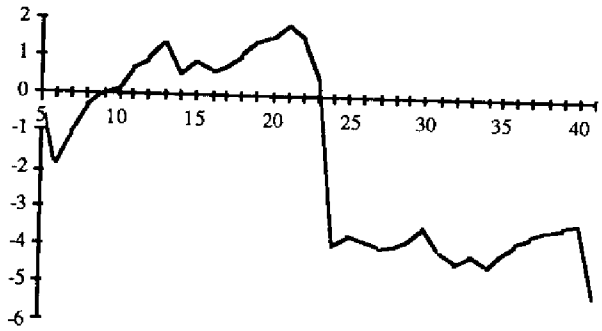


Figure III-9 : Test de la moyenne arithmétique pour les données du système 4

Le système 9 quant à lui présente deux phases bien distinctes : une phase de légère décroissance suivie d'une phase de forte croissance de fiabilité (figure III-10a). Ceci est confirmé par la courbe de la figure III-10b qui donne le coefficient de Laplace pas à pas et sur laquelle on voit bien l'inversion vers les données 23-24.



a- Moyenne arithmétique



b- Coefficient de Laplace

Figure III-10 : Tests de tendance pour les données du système 9

Il est intéressant de remarquer que bien que les systèmes 2 et 9 aient un coefficient de Laplace du même ordre de grandeur, l'allure de la croissance est différente.
Pour le système 9, quatre données ont été identifiées comme douteuses (pour un total de 41) correspondant respectivement aux numéros : 41, 24, 31 et 34 ; après suppression de ces données douteuses, la courbe donnant le nombre moyen de défaillances par périodes de 5 jours a effectivement la forme d'une cloche.

Pour certains systèmes tels que les systèmes 6, 7, 11A, 11B, 12, 13 et 14 (figure III-11) on observe un phénomène d'oscillation. Ces systèmes peuvent être modélisés par un processus de Poisson homogène (HPP). Le système 7 n'a un coefficient de tendance inférieur à -2 que pour les trois dernières données dont deux ont été trouvées douteuses. Le système 12 qui est le seul système à avoir un coefficient de tendance global (légèrement) positif, n'a eu un coefficient négatif qu'exceptionnellement et jamais inférieur à -1.6. Lui aussi peut très bien être modélisé par un HPP.

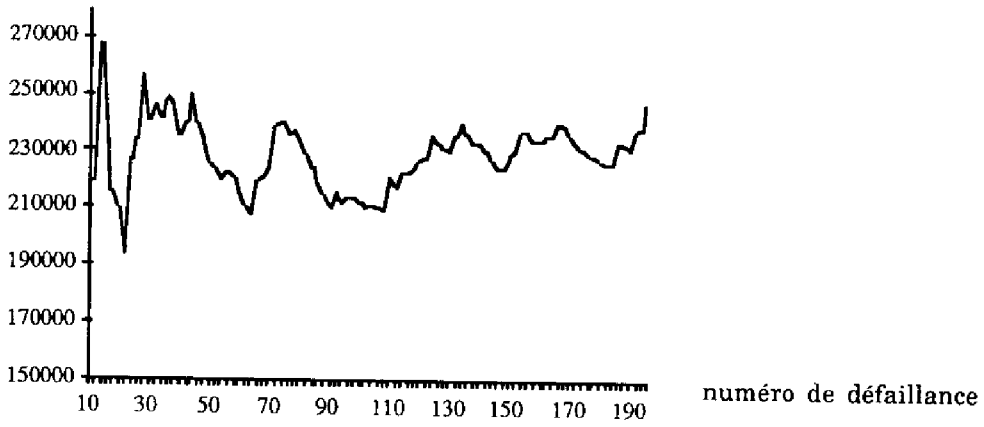


Figure III-11: Test de la moyenne arithmétique pour le système 14

III-4-4 Conclusions

Les traitements effectués sur les données RADC dans ce paragraphe montrent que l'étude de fiabilité peut être arrêtée là pour certains jeux de données tels que ceux correspondant aux systèmes 6, 7, 11A, 11B, 12, 13 et 14.
Pour ces données, il n'y a pas croissance de fiabilité : l'évaluation du taux de défaillance peut alors être réalisée à l'aide d'un modèle exponentiel (le processus de défaillance est, dans ce

cas, considéré comme HPP) ; le MTTF estimé de la i ième défaillance est alors donné par la moyenne arithmétique des observations d'ordre inférieur à i. Pour les autres systèmes, il y a croissance de fiabilité globale malgré quelques fluctuations pour certains tels les systèmes 8 et 9.

Le manque d'informations concernant les systèmes étudiés ne nous a pas permis d'interpréter convenablement les résultats issus des différents tests. La connaissance des fonctions des systèmes, des conditions de collecte des données et de l'environnement d'utilisation est nécessaire pour mener à bien cette étape ce qui confirme la nécessité d'une approche plus globale de la fiabilité du logiciel. Ceci sera confirmé par l'étude des autocommutateurs téléphoniques (chapitre V) où notre connaissance des systèmes et de leur utilisation est meilleure et a permis de mener l'approche globale que nous proposons.

III-5- CONCLUSIONS

Ce chapitre a été consacré au prétraitement des données : nous avons présenté et discuté deux types de tests à appliquer aux données collectées sur un système avant d'évaluer la fiabilité du logiciel à l'aide des modèles de croissance de fiabilité : le test des données douteuses et les tests de tendance. L'application conjointe de ces tests à des ensembles de données recueillies sur des systèmes réels a montré la nécessité de bien connaître les systèmes étudiés afin de pouvoir interpréter correctement les résultats et de prendre une décision en ce qui concerne l'élimination des données identifiées comme douteuses.

L'appréciation de l'évolution de la fiabilité est déterminante pour l'application des modèles de croissance de fiabilité : il faut bien connaître l'allure de l'évolution de la fiabilité pour localiser les périodes de fluctuation de la fiabilité. Les données sont ainsi réparties en paliers correspondant soit à une croissance de fiabilité soit à une décroissance-croissance afin de choisir à la fois les modèles les mieux appropriés et les données à fournir à ces modèles.

Le prétraitement des données peut ainsi constituer à lui seul la deuxième étape de l'étude de fiabilité, notamment quand on cherche à disposer d'éléments concrets de suivi de la fiabilité.

CHAPITRE IV

APPLICATION DES MODELES

DE CROISSANCE DE FIABILITE

INTRODUCTION

Les tests effectués dans le chapitre précédent permettent d'une part de suivre l'évolution de la fiabilité du logiciel et de prendre une décision en ce qui concerne le(s) modèle(s) de croissance de fiabilité à utiliser pour l'évaluation de la fiabilité du logiciel d'autre part. A partir de ces modèles on peut évaluer l'ensemble des mesures de la sûreté de fonctionnement du logiciel présentées au premier chapitre. Bien que ces mesures soient liées, certaines sont plus significatives que d'autres selon le but recherché, ce but étant lui-même fonction à la fois des objectifs de l'étude, de la phase du cycle de vie concernée par l'évaluation et des spécifications du système. Il n'est donc pas nécessaire d'évaluer systématiquement toutes les mesures mais il faut répondre à la question qui se pose : quelles mesures évaluer et sous quelle forme présenter les résultats de l'évaluation de ces mesures ?

Une fois ces mesures choisies il faut :
- prendre une décision en ce qui concerne la façon d'appliquer le(s) modèle(s) : utiliser toutes les données, ou partitionner les données selon les différentes phases du cycle de vie ou selon les modes de défaillance,...
- calibrer le(s) modèle(s) (c'est-à-dire estimer les paramètres) à l'aide des données précédentes afin d'évaluer les différentes mesures de la sûreté de fonctionnement,
- s'assurer de la validité des résultats fournis par le(s) modèle(s) à l'aide d'un ou de plusieurs critères de validation et de comparaison de modèles.

Tous ces aspects sont étroitement liés : le choix des mesures à évaluer conditionne la variable aléatoire à prendre en compte pour les évaluer (donc l'ensemble des données à utiliser) et le choix de la variable aléatoire induit automatiquement un choix au niveau de la méthode de calibrage du modèle et des méthodes de validation de ce dernier.

Le but de ce chapitre est de traiter de l'ensemble des aspects liés à l'application des modèles de croissance de fiabilité. Il est divisé en trois paragraphes :

- le premier est consacré à une discussion sur le choix des mesures de la sûreté de fonctionnement à évaluer, la façon de les présenter ainsi que la détermination des données à utiliser pour les obtenir,
- le deuxième traite de l'application des modèles par composant et par conséquence des défaillances,
- le troisième traite des aspects de calibrage et de validation des modèles.

IV-1- CHOIX DES MESURES ET DES DONNEES

IV-1-1 Choix des mesures

Le but d'une évaluation de fiabilité dépend de la phase pour laquelle l'évaluation est effectuée.

En développement, **les différents intéressés** veulent :
- s'assurer, avant même de commencer la conception, de la faisabilité en ce qui concerne les clauses de sûreté de fonctionnement du système,
- répercuter ces clauses au niveau des différentes entités constituant le système et faire un choix d'architecture matérielle et logicielle permettant de respecter les clauses de sûreté de fonctionnement globale du système,
- avoir des éléments de suivi de l'avancement du développement, à la fois du processus de développement lui-même et du produit logiciel,
- s'assurer que les objectifs de qualité et de fiabilité sont atteints avant livraison du système.

En vie opérationnelle :
- **l'utilisateur** désire avoir des renseignements concernant le taux de défaillance résiduel, le coût entraîné par la perte de service, la fréquence des appels à maintenance ; il souhaite surtout une évaluation de l'impact des fautes résiduelles sur la sûreté de fonctionnement de son système afin de s'assurer que la sûreté de fonctionnement du système est conforme aux objectifs exprimés dans les spécifications,
- **le concepteur** est plutôt intéressé par l'estimation de l'effort de maintenance qu'il faut continuer à assurer au niveau de toutes les instances du logiciel en utilisation sur les différents sites.

L'évaluation de la fiabilité d'un logiciel doit donc s'intégrer dans le processus d'évaluation de la sûreté de fonctionnement du système dont le logiciel fait partie. Les objectifs de sûreté de fonctionnement sont généralement exprimées sous l'une des formes suivantes :
- taux de défaillance maximal admissible pour le système ou taux de défaillance nominal (ou intervalle entre défaillances minimal ou nominal), c'est le cas par exemple des applications

aéronautiques où l'objectif est d'atteindre un taux de défaillance inférieur à 10^{-9}/h pour des durées de mission de 10 heures [Wen 78, Lal 88],
- indisponibilité maximale admissible pour le système, à titre d'exemple, les objectifs de sûreté pour le système informatique de localisation du programme SARSAT-COSPAS sont d'assurer une disponibilité moyenne de 0,99 [Des 82].
A partir des objectifs de sûreté de fonctionnement du système, il faut exprimer ceux du logiciel. Il est donc logique d'exprimer ces derniers dans les mêmes termes notamment en ce qui concerne la vie opérationnelle.

Durant les premières phases du développement le profil d'utilisation étant très différent de celui de la vie opérationnelle, évaluer un taux de défaillance ne peut pas être très significatif pour un utilisateur potentiel. On cherchera donc à suivre l'évolution de certaines mesures : soit le nombre cumulé de défaillances, soit les intervalles entre défaillances. L'estimation du nombre de défaillances susceptibles d'avoir lieu sur la période suivante constitue une indication sur le nombre de corrections qu'il faudrait prévoir durant cette période.

A la fin du développement, le profil d'utilisation se rapprochant de celui de la vie opérationnelle, grâce à l'utilisation de simulateurs d'environnement, les données de défaillance peuvent servir à prévoir le comportement du logiciel en opération (moyennant quelques extrapolations), et dans ce cas on peut évaluer soit la disponibilité du système soit un MTTF ou un taux de défaillance, si on est intéressé par la fiabilité. Si en revanche, on veut estimer l'effort de maintenance qu'il faut prévoir en vie opérationnelle il faudra évaluer le nombre de défaillances qui sont susceptibles d'avoir lieu afin de mettre en place le potentiel humain et matériel nécessaire pour l'assurer convenablement.

En vie opérationnelle, on évaluera donc plutôt des mesures du type MTTF et taux de défaillance (ou intensité de défaillance).

Avant d'analyser plus en détail les mesures à évaluer, nous pensons qu'il est nécessaire de faire quelques commentaires concernant une mesure couramment évaluée en fiabilité du logiciel : le nombre de fautes résiduelles. Ce nombre est lié aux caractéristiques de $(E_e \cup E_{af})$; plus il est petit, meilleure est le programme. On pourrait penser qu'il constitue une bonne mesure de la fiabilité. En fait si ce nombre est "nul" le logiciel est fiable, mais un logiciel ayant un bon niveau de fiabilité n'a pas forcément un nombre de fautes résiduelles très faible, de même, un logiciel contenant un petit nombre de fautes qui sont souvent activées n'est pas fiable.

Pour illustrer ceci nous nous basons sur les résultats publiés dans [Ada 80] où neuf logiciels ont été analysés. Ces logiciels ont été choisis dans la gamme IBM et correspondent à des programmes de volume important utilisés un peu partout dans le monde et sur lesquels un très grand nombre de défaillances ont été observées. La figure IV-1 donne, pour chaque logiciel, le pourcentage de fautes ayant entraîné un intervalle entre défaillances (MTTF) compris entre celui indiqué par la colonne précédente et celui indiqué par la colonne correspondante ; par exemple pour le logiciel numéro 1, 0,7% des fautes ont entraîné un MTTF inférieur à 1,58 an et 1,2% des fautes ont entraîné un MTTF compris entre 1,58 an et 5 ans.

Il est intéressant de remarquer que, bien que les logiciels soient différents, ces chiffres sont tout à fait comparables entre eux. On peut aussi remarquer que 2% des fautes (les deux premières colonnes) ont entraîné à elles seules 1000 fois plus de défaillances que 60% des fautes : celles relatives aux deux dernières colonnes. En supposant que la durée de vie d'un logiciel est de l'ordre de 15 ans, ces chiffres montrent que uniquement près de 5% des fautes vont se manifester durant la vie du logiciel : celles correspondant à la somme des trois premières colonnes.

MTTF (ans) / Logiciel	1,58	5	15,8	50	158	500	1580	5000
1	0,7	1,2	2,1	5,0	10,3	17,8	28,8	34,2
2	0,7	1,5	3,2	4,5	9,7	18,2	28,0	34,3
3	0,4	1,4	2,8	6,5	8,7	18,0	28,5	33,7
4	0,1	0,3	2,0	4,4	11,9	18,7	28,5	34,2
5	0,7	1,4	2,9	4,4	9,4	18,4	28,5	34,2
6	0,3	0,8	2,1	5	11,5	20,1	28,2	32,0
7	0,6	1,4	2,7	4,5	9,9	18,5	28,5	34,0
8	1,1	1,4	2,7	6,5	11,1	18,4	27,1	31,9
9	0,0	0,5	1,9	5,6	12,8	20,4	27,6	31,2

Figure IV-1 : Pourcentage de fautes et MTTF correspondant

Sans aller jusqu'à dire que ces chiffres sont applicables à tout logiciel, nous pouvons en déduire que le nombre de fautes résiduelles n'est pas du tout significatif d'autant plus que ce nombre ne peut "jamais" être connu.

Par contre, comme nous l'avons déjà vu, le nombre de défaillances ayant eu lieu constitue un bon indicateur de l'évolution du développement. De façon plus précise, au cours du développement, le logiciel subit un certain nombre de tests : des tests unitaires[4] , des tests d'intégration et des tests de validation. Pour chacun de ces tests, on prélève des données

[4] Le test unitaire peut correspondre soit au test d'une fonction élémentaire du logiciel soit au test d'une fonction du système elle-même composée de fonctions élémentaires.

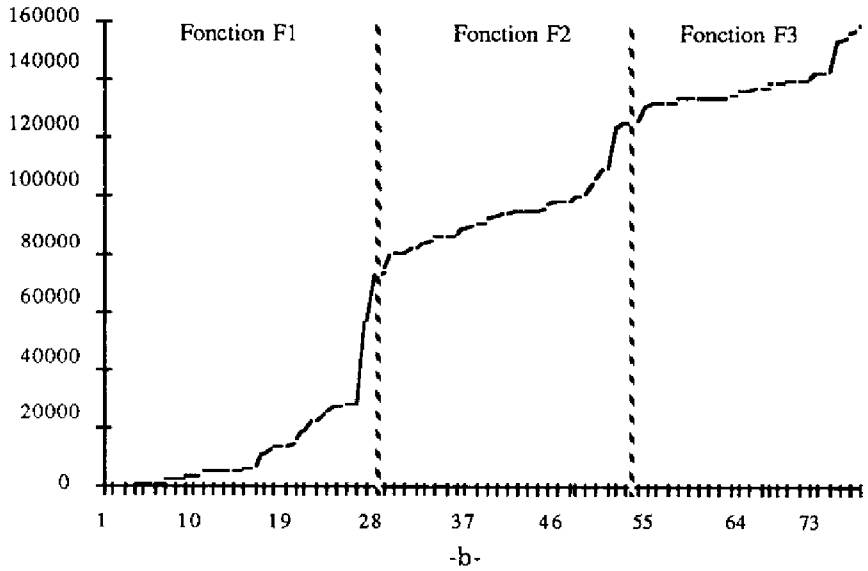
auxquelles on applique des modèles ; le suivi de l'évolution du nombre de défaillances constitue un bon indicateur de l'évolution de la fiabilité de ce logiciel vis-à-vis de l'ensemble des tests de la phase concernée et peut contribuer à la décision de passer aux phases suivantes. La prévision du nombre de défaillances qui auront lieu durant le période suivante peut servir de base à la planification du nombre de corrections qui devraient être effectuées, ce qui dimensionne l'effort (en temps et en nombre de personnes) qu'il est nécessaire de mettre en place.

Le cas du logiciel étudié dans [Cur 86] constitue un bon exemple. Il s'agit d'un logiciel qui a été développé progressivement en testant d'abord une première fonction F1 toute seule puis en ajoutant une deuxième fonction F2, puis une troisième F3. Lors du test de F1, 27 défaillances ont été enregistrées ; après intégration de F2, un test de régression a permis d'activer une faute dans F1 et lors de l'intégration de F3 deux fautes ont été activées dans F1. Pour F2, 20 défaillances ont eu lieu lors du test de F2, et 5 lors du test de F3. 21 défaillances ont eu lieu dans F3 lors du test de F3. Les intervalles entre défaillances sont donnés sur la figure IV-2-a et le tracé du temps cumulé de fonctionnement en fonction du nombre cumulé de défaillances N(t) est donné par la figure IV-2-b.
N(t) croît rapidement lors du test de F1 pour atteindre une certaine stabilité à la fin du test de F1, stabilité perturbée par le test de F2 mais qui apparaît à nouveau jusqu'au début du test de F3 ; à la fin du test de F3 la stabilisation de N(t) n'est pas aussi nette. La stabilisation de N(t) indique que l'ensemble des tests appliqués à la fonction en cours d'étude a permis de révéler la majorité des fautes pour lesquelles il a été conçu et qu'il a atteint ses limites.
Les données de la figure IV-2 ne permettent pas d'évaluer la fiabilité du logiciel en opération, elles constituent néanmoins de bonnes bases pour une réflexion sur l'évolution du projet.

Ces chiffres constituent une bonne illustration de quelques points énoncés dans les deux premiers chapitres :
- les intervalles entre défaillances dont la moyenne en fin de test de F1 est de l'ordre de 2000 secondes (à l'exclusion des données relatives aux tests de régression), sont à peu près maintenus au début du test de F2 mais décroissent tout de suite après ; ce phénomène est plus marqué au niveau du passage entre le test de F2 et F3 où la moyenne passe de 1570 secondes pour les cinq dernières données relatives à F2 (sans compter les tests de régression) à 625 s pour les cinq premières de F3,
- les courbes N(t) relatives à F2 et F3 ont des points d'inflexion et confirment le fait qu'à l'intérieur du test d'une même unité on peut avoir une décroissance de fiabilité avant d'avoir la croissance souhaitée.

Fonction F1	Fonction F2	Fonction F3
85	117	1129
85	2247	72
479	1813	597
965	2014	394
469	341	277
385	842	394
796	1301	213
277	1940	938
927	646	1012
340	1460	788
405	1151	1161
150	1130	277
277	213	181
503	554	767
694	1364	1086
620	1321	394
4050	213	1608
3064	1609	593
522	725	11075
1797	3985	2157
2329	5585	1448
3798	13120	
2775	1524	
2393	362	
909	4794	
994		
28212		
14956		
1971		
5308		

-a-



-b-

Figure IV-2 : Intervalles entre défaillances pour les trois fonctions du système

Ce raisonnement peut être poursuivi : puisque l'évolution de N(t) constitue un indicateur de l'évolution des phases de test, on peut anticiper et se servir de l'estimation par les modèles de N(t) (c'est-à-dire H(t)) pour évaluer le temps au bout duquel il faut passer aux tests suivants ou à la phase suivante. On utiliserait les premières données issues du test pour estimer les paramètres du modèle et tracer H(t) afin d'évaluer approximativement l'instant de début de stabilisation, ce qui constitue une indication du temps au bout duquel le test aurait atteint ses limites. Il faudrait appliquer les modèles en parallèle avec le test afin de pouvoir influencer le déroulement du test. Les résultats ne sont fiables que si l'effort de test ne varie pas considérablement le long de cette phase et, même dans ces conditions, ils doivent être, non pas utilisés comme des mesures de fiabilité, mais interprétés comme des indicateurs. Une telle démarche a été suivie lors de l'étude de la fiabilité de l'autocommutateur téléphonique TROPICO-R [Kan 88d] et donne, comme nous le verrons dans le dernier chapitre de ce mémoire, des résultats tout à fait comparables à ceux observés par la suite sur le système.

L'utilisation "habituelle" de H(t) consiste à se fixer un nombre de défaillances a priori (qui correspond généralement au nombre de fautes résiduelles dans le programme en début de période de test estimé par le modèle) et d'évaluer le temps au bout duquel ce nombre sera atteint. Les résultats obtenus par les modèles de croissance de fiabilité appliqués de cette manière ne sont pas généralement très convaincants [Mus 79], à moins de se servir des résultats des tests de tendance pour appliquer les modèles de croissance de fiabilité sur les périodes délimitées par ces tests.

D'après ce qui précède nous voyons que deux mesures de sûreté de fonctionnement semblent intéressantes à évaluer aussi bien en développement qu'en vie opérationnelle : elles correspondent au MTTF et au taux de défaillance. L'intensité de défaillance sera obtenue à partir du tracé des taux de défaillance successifs. Le suivi de l'évolution du nombre de défaillances observées constitue à la fois un bon indicateur du sens de variation de la fiabilité et un moyen d'aide à la planification de l'effort de test et de maintenance. Le choix des formes sous lesquelles les résultats sont présentés est considéré dans le paragraphe suivant.

IV-1-2 Présentation des mesures

Evaluer un taux de défaillance ou un MTTF revient à donner une estimation ponctuelle à laquelle on peut associer des intervalles de confiance ou plus exactement des quantiles d'ordre α et $(1-\alpha)$.

Pour une distribution F(x), le quantile d'ordre α est la racine de l'équation :

$$F(x) = \alpha \qquad 0 < \alpha < 1$$

Lien entre quantile d'ordre α et niveau de confiance α : la borne supérieure de confiance d'une variable correspond au quantile d'ordre α de la distribution et la borne inférieure de confiance correspond au quantile d'ordre $(1-\alpha)$ de la distribution.

Si x_{sup} et x_{inf} sont respectivement les limites de confiance supérieure et inférieure on a :

$P(x \leq x_{sup}) = \alpha$

et $P(x >= x_{inf}) = \alpha$ ou, ce qui est équivalent, $P(x \leq x_{inf}) = 1- \alpha$

Pour un niveau de confiance $\alpha = 0.9$ il faudrait évaluer aussi le quantile d'ordre 0.1, pour le niveau $\alpha = 0.8$, il faudrait évaluer le quantile d'ordre 0.2 ; on voit que plus α est grand plus l'intervalle entre les deux bornes est grand et moins la notion de niveau de confiance est intéressante. Si on prend des niveaux de confiance de plus en plus grands l'intervalle entre les deux bornes est de plus en plus réduit mais on perd aussi l'avantage des niveaux de confiance. Les niveaux de confiance pour le MTTF, dans le contexte croissance de fiabilité du logiciel ont été utilisés dans [Mus 79, Fav 85]. Nous avons constaté de façon générale que les limites de confiance ne sont rapprochées que lorsque les données collectées sont assez régulières : croissance monotone ou stabilisée ; dans le cas où les données sont moins régulières, les intervalles entre les deux bornes sont très grands et donc peu significatifs. En fait, quand les données sont régulières, les modèles de croissance de fiabilité donnent de bons résultats aussi, et dans ce cas l'utilité des bornes de confiance est limitée.

Il faut cependant préciser que la borne inférieure de confiance revêt une importance particulière dans certains cas, notamment quand les spécifications du système (et plus particulièrement celles du logiciel) sont exprimées sous la forme :

"le MTTF doit être supérieur à T_0 avec un risque au plus égal à β".

Ceci signifie que le quantile d'ordre β (ou la borne inférieure de confiance au niveau $\alpha=1-\beta$) doit être égal à T_0. On doit donc s'assurer lors de l'évaluation de la fiabilité en vie opérationnelle que cet objectif est bien atteint.

La limite des niveaux de confiance correspond au cas $\alpha = 0.5$, ce qui revient à évaluer la médiane de la distribution. Ceci a été utilisé dans certaines publications [Abd 86, Cha 86]. A notre avis il n'y a pas de différence significative entre la médiane et le MTTF qui est la moyenne de la distribution, la seule différence est d'ordre pratique : pour certains modèles il n'est pas possible d'obtenir l'expression du MTTF alors que l'évaluation de la médiane est toujours possible. La question ne se pose même pas si l'objectif de l'étude de fiabilité est d'évaluer un MTTF.

Ce que nous venons de voir pour le MTTF est aussi valable pour le taux de défaillance à condition de remplacer borne inférieure de confiance par borne supérieure de confiance : il faut garantir un taux de défaillance inférieur au quantile d'ordre α, le risque étant égal à $\beta=1-\alpha$.

Nous évaluerons donc des estimations ponctuelles du MTTF ou du taux de défaillance. Cependant afin d'indiquer la "distance" entre la valeur observée de l'intervalle entre défaillances et celle estimée par la distribution il est intéressant d'évaluer la fonction de répartition à l'instant t_i (t_i est la valeur observée de l'intervalle entre défaillances) :

$$u_i = F(t_i) = P(T_i \leq t_i).$$

L'évaluation de u_i est à la base de l'une des méthodes de validation des modèles de croissance de fiabilité : la distance de Kolmogorov-Smirnov.

Remarque

Les notions de borne supérieure et inférieure de confiance telles qu'elles ont été définies par rapport au MTTF ne doivent pas être confondues avec celles définies par rapport à la valeur la plus probable d'une distribution. Lorsque la fonction densité de probabilité a un maximum (distribution unimodale), il existe une région autour de laquelle la variable aléatoire a plus de chance de se réaliser. Par rapport à cette région, on peut définir un intervalle de confiance et des bornes de confiance. Ceci est couramment utilisé en statistique et pourrait aussi être utilisé dans ces études, mais il ne faut surtout pas confondre la valeur la plus probable avec le MTTF en tant que mesure de la sûreté de fonctionnement (qui correspond à la moyenne de la variable aléatoire). Nous préférons cependant évaluer le MTTF, d'autant plus que toutes les distributions n'ont pas forcément un maximum et que, souvent, les spécifications sont exprimées en terme de MTTF.

IV-1-3 Choix des données

Dans le cas où l'on est intéressé par l'évolution de la fiabilité du produit (sur plusieurs versions successives) et qu'il y a croissance de fiabilité globale, on peut appliquer les modèles de croissance de fiabilité à toutes les données. Les résultats obtenus constituent un lissage de la courbe intensité de défaillance.

Si on veut avoir des résultats concernant une version spécifique du programme ou une phase particulière du cycle de vie en utilisant ces modèles il faut travailler sur des données qui suivent une de ces allures. Il faut donc regrouper les données recueillies sur le logiciel par phase ou par version afin d'avoir des évaluations à partir de données plus homogènes ; on améliore ainsi la qualité de l'évaluation.

De même, on peut évaluer la fiabilité du logiciel selon plusieurs modes de défaillances dans le but d'évaluer l'importance des défaillances les plus graves ; il faut alors regrouper les données relatives à ces modes et appliquer les modèles aux différents groupes.

Enfin un troisième type de regroupement peut être effectué : regroupement par composants afin d'évaluer l'impact de chaque composant logiciel sur la fiabilité globale du système et d'identifier les composants les moins fiables.

Les modèles de croissance de fiabilité seront appliqués séparément sur chacun de ces groupes. Cependant, nous avons vu au chapitre II que, bien que les modèles de connaissance que nous avons établis permettent de suivre n'importe quelle évolution de la fiabilité, les modèles d'action dont nous disposons actuellement sont capables de modéliser soit une croissance de fiabilité soit une décroissance suivie d'une croissance de fiabilité (modèles en forme de S). Il faut donc appliquer ces modèles sur des données dont l'évolution correspond à une croissance ou décroissance-croissance. Ceci amène à partitionner les données d'un groupe selon les périodes de croissance et de décroissance-croissance de fiabilité, de façon à appliquer un modèle en fonction des hypothèses selon lesquelles il a été conçu ; ces périodes correspondent aux paliers définis au chapitre III. On améliore ainsi la qualité de l'évaluation [Sab 86, Kan 87c, Bas 88a,b].

Si :
- les données disponibles correspondent à plusieurs phases du cycle de vie et les limites entre ces différentes phases ne sont pas répertoriées,
- et qu'on n'arrive pas à faire apparaître clairement des paliers de croissance et de décroissance de fiabilité,
on ne peut faire de partition nette des données et on sélectionnera alors progressivement des ensembles de données voisines dans le temps à l'aide d'une **fenêtre**.

Le principe de l'application par fenêtre est le suivant : on se fixe une longueur de fenêtre en fonction du nombre total de données disponibles et de la tendance indiquée par les tests. Les modèles de croissance de fiabilité seront appliqués en utilisant le même nombre de données (celui correspondant à la longueur de la fenêtre) ; à chaque fois que l'on ajoute une nouvelle donnée, on retranche la première donnée (dans l'ordre chronologique). Un exemple d'application des modèles de croissance de fiabilité par fenêtre se trouve dans [Kan 87a].

En résumé, trois types de regroupements des données sont possibles : par phase ou par version, selon les différents modes de défaillance et selon les composants logiciels.

A l'intérieur de chaque groupe les données seront réparties par palier ou par fenêtre selon les résultats des tests de tendance.

IV-1-4 Conclusions

Ce paragraphe a permis de retenir trois mesures de base à évaluer par les modèles de croissance de fiabilité selon la phase du cycle de vie concernée et l'objectif de l'évaluation : le temps moyen entre défaillances (MTTF) des défaillances futures, le taux de défaillance (ou l'intensité de défaillance) et le nombre de défaillances. Nous recommandons aussi d'évaluer le quantile d'ordre (1-α) qui constitue la borne de confiance inférieure au niveau α, afin de "garantir" avec un risque α que le MTTF sera supérieur à cette borne.

En ce qui concerne les données à utiliser pour l'évaluation de la fiabilité du logiciel, les données collectées sur le logiciel peuvent être regroupées par phase (ou par version), par composant ou selon les modes de défaillance. Dans chaque groupe une répartition temporelle peut être effectuée. Cette répartition temporelle est objective et facile si on dispose de données précises concernant les phases du cycle de vie et le profil d'utilisation, mais plus subjective et difficile dans le cas contraire où une large part est laissée à l'initiative de la personne qui effectue l'évaluation en fonction des résultats des tests de tendance.

IV-2- COMPOSANTS ET CONSEQUENCES

Ce paragraphe est consacré à l'étude de quelques points relatifs à l'application des modèles de croissance de fiabilité par composant et par conséquence.

IV-2-1 Composants

IV-2-1-1 Intérêts de la répartition par composants

Les modèles de croissance de fiabilité ne tiennent pas compte de la structure interne du logiciel : celui-ci est considéré comme une boîte noire sur laquelle on fournit des données et on observe les sorties. Dans le cas où le programme est constitué de composants, il est intéressant d'identifier le composant qui est à l'origine de la défaillance. L'évaluation de la fiabilité de chacun des composants permet :

- d'identifier les composants les moins fiables afin de mettre l'accent sur les composants de même nature lors du développement de futurs systèmes semblables,
- d'évaluer l'impact des différents composants logiciels sur la fiabilité globale du logiciel ainsi que sur celle du système.

IV-2-1-2 Choix des composants

La décomposition est généralement guidée par les fonctions que doit accomplir le système : un composant peut correspondre par exemple à un traitement élémentaire ou à une tâche (groupe de traitements élémentaires).

Le choix de la taille moyenne des composants à considérer (et donc du nombre de composants) résulte d'un compromis entre le nombre de données disponibles pour chaque composant et le niveau de détail auquel on veut modéliser le système. Une décomposition trop fine du logiciel peut entraîner un petit nombre de données de défaillance pour les composants et rendre difficile une étude de fiabilité par composant. A l'inverse une décomposition en peu de composants peut ne pas donner des indications intéressantes sur la fiabilité du logiciel car elle pourrait masquer certains phénomènes qui ne pourraient être mis en évidence qu'avec une décomposition plus fine.

IV-2-1-3 Lien intensité de défaillance du logiciel — taux de défaillance des composants

Afin d'étudier l'influence des corrections apportées aux différents composants sur la fiabilité du logiciel nous commencerons par établir le lien entre le taux de défaillance des différents composants et celui du logiciel entre deux défaillances.

Taux de défaillance :

Dans le cas où l'on est intéressé par le taux de défaillance, le modèle structurel présenté dans [Lit 79] ou celui présenté dans [Lap 88] s'appliquent. Ce dernier a l'avantage de permettre de modéliser le fait que plusieurs composants peuvent être en cours d'exécution simultanément. Il peut être résumé comme suit : le taux de défaillance λ du logiciel entre deux défaillances est constant et est égal à :

$$\lambda = \sum_{k=1}^{Cp} \pi_k \lambda_k \qquad \text{(IV-1)}$$

où :

- Cp est le nombre total de composants,
- π_k la proportion moyenne du temps où le composant k est en cours d'exécution en l'absence de défaillance,
- λ_k le taux de défaillance du composant k.

Les hypothèses nécessaires pour l'établissement de cette relation sont :
- l'indépendance des différents composants vis-à-vis du processus de défaillance,
- un grand nombre de transitions relatives au processus d'exécution avant défaillance.

Le terme $\pi_k \lambda_k$ correspond au taux de défaillance du composant dans son environnement d'application et est appelé taux de **défaillance apparent**.
Le taux de défaillance du logiciel est donc égal à la somme des taux de défaillance apparents de ses différents composants. Sous cette forme simplifiée le modèle aurait pu être obtenu en considérant que le processus de défaillance du logiciel résulte de la superposition des processus de défaillance apparents de ses composants. Le processus de défaillance de chaque composant étant un processus de Poisson homogène, le processus de défaillance résultant est aussi un processus de Poisson homogène dont le taux est égal à la somme des taux de ses constituants [Bar 75].
L'intensité de défaillance pour un composant peut alors être établie à partir de son taux de défaillance apparent.

Intensité de défaillance du logiciel :

Après la défaillance de l'un quelconque de ses composants, la correction introduite au niveau du logiciel modifie le (ou les) taux de défaillance du (ou des) composant(s) concerné(s). L'expression IV-1 est encore valable, avec toutefois :
- des taux de défaillance des différents composants (λ_k) modifiés,
- des π_k qui pourraient éventuellement changer.

Si on suppose qu'après correction, le logiciel est placé à nouveau dans des conditions de sollicitation ne différant pas des précédentes, les proportions moyennes du temps d'exécution des différents composants (π_k) sont inchangées ; ceci correspond aux dernières phases du développement et surtout à la vie opérationnelle.
On voit que la variation du taux de défaillance après correction est directement liée à celle de ses composants. Après la (i-1) ième correction le taux de défaillance est donné par :

$$\lambda_i = \sum_{k=1}^{Cp} \pi_k \lambda_{ik} \qquad \text{(IV-2)}$$

où λ_{ik} est le taux de défaillance du composant k après (i-1) corrections du logiciel.

En supposant qu'il y a correction après chaque défaillance, l'expression de l'intensité de défaillance du logiciel est donnée par les expressions II-7 à II-8 établies au chapitre II en remplaçant les λ_i par leurs expressions IV-2. De façon plus générale, en supposant des politiques de correction telles que celles étudiées au second chapitre, les expressions établies restent également valables à condition de remplacer les λ_i par leurs expressions.

Nous avons vu que l'allure de l'évolution de l'intensité de défaillance est directement liée à l'allure de l'évolution des taux de défaillance, ce taux étant lui même directement lié aux taux des différents composants. On en déduit que :

- si les processus de défaillance de chacun des composants est à taux strictement décroissant, le processus résultant est à intensité strictement décroissante,
- si un des processus de défaillance est à taux non strictement décroissant mais globalement décroissant, la croissance locale se retrouve au niveau du processus global mais elle est atténuée (ou accentuée) par la décroissance (respectivement, croissance) des taux des autres composants.

Fiabilité stabilisée :

En fiabilité stabilisée les taux de défaillance des différents composants sont inchangés puisqu'on n'introduit plus de corrections ou les corrections introduites ne changent pas de façon significative ces taux et les conditions de sollicitation atteignent elles aussi un régime stabilisé ; il s'en suit que le taux de défaillance du logiciel est constant. L'intensité de défaillance du logiciel est donc égale à ce taux ; elle est donc égale à la somme des intensités de défaillance de ses composants en régime stabilisé.
Les taux de défaillance en fiabilité stabilisée peuvent être évalués à l'aide du modèle hyperexponentiel.

Remarque :
Pour le modèle présenté dans [Lit 79], l'expression du taux de défaillance du composant k est de la forme :

$$\lambda_k = \nu_k + \sum_{i \neq k} q_{ki}\, \beta_{ki}, \qquad \text{(IV-3)}$$

les ν_k étant les taux de défaillance internes aux composants, les β_{ki} les probabilités de défaillance aux interfaces et q_{ki} le taux de transition du composant k vers le composant i.
Le raisonnement suivi en utilisant IV-1 reste valable dans ce cas aussi, avec cependant une complexité supplémentaire pour l'estimation des paramètres.

IV-2-2 Conséquences

IV-2-2-1 Intérêt du regroupement par conséquences

La prise en compte de toutes les sources de défaillance et de tous les modes de défaillance permet d'avoir des renseignements sur le nombre moyen d'appels à maintenance ou de réinitialisations du programme. Toutefois l'impact des défaillances sur le service délivré par le système n'est pas le même et les conséquences des perturbations sur l'environnement n'ont pas la même envergure.
Pour un réseau téléphonique, un appel qui n'aboutit pas a moins d'importance qu'une coupure de communication déjà établie qui est elle même moins grave que la perte de tout l'organe de commutation.

Pour un système de contrôle au sol d'un satellite assurant des fonctions de suivi du satellite et d'archivage des données en provenance de ce satellite, il est moins critique de perdre momentanément les fonctions d'archivage que le suivi du satellite dont la trajectoire pourra s'écarter de la trajectoire nominale si le suivi est interrompu durant plus d'un quart d'heure. Il est donc intéressant d'évaluer la sûreté de fonctionnement d'un système selon les différentes conséquences (ou modes de défaillance). Au niveau du logiciel, il faut regrouper l'ensemble des données collectées en autant de sous-ensembles que de modes de défaillance recensés. L'application des modèles de croissance de fiabilité se fait sur ces sous-ensembles de la même manière que pour l'ensemble des données et permet d'évaluer les taux de défaillance relatifs à chacun des modes de défaillance.

IV-2-2-2 Identification des conséquences

En appliquant les modèles de croissance de fiabilité selon les modes de défaillance, on rencontre trois types de difficulté :
- recenser les modes de défaillance du système,
- établir le lien entre modes de défaillance du système et mode de défaillance du logiciel,
- établir la correspondance entre un mode de défaillance et les données collectées sur le logiciel qui lui sont associées.

Le nombre de modes de défaillance à prendre en compte est fonction du niveau de détail auquel on veut modéliser le comportement du système : plus la décomposition est fine plus la précision de l'évaluation est grande et plus la modélisation du système (matériel et logiciel) est complexe. Considérer un grand nombre de modes de défaillance entraîne une répartition des défaillances collectées en autant de sous-ensembles : il y a donc un compromis à faire entre le niveau de détail souhaité et le nombre de modes de défaillance à considérer. Une répartition en trois ou quatre conséquences est généralement suffisante : un mode de défaillance critique (grave ou catastrophique), un mode de défaillance bénigne et un ou deux modes de dégradation de service regroupant les différents types de conséquences les plus significatives. Une répartition en deux modes peut suffire dans le cas où l'utilisateur est surtout intéressé par le taux des défaillances entraînant une indisponibilité du système et le taux global de défaillance. En fait les modes de défaillance sont aussi fonction des durées de l'arrêt du système pour relance. Comme nous l'avons vu au chapitre II, ces durées interviennent explicitement pour l'évaluation de la disponibilité du système.

Le lien mode de défaillance système—mode de défaillance logiciel doit être établi de façon précise et ne peut être étudié de façon générale. Il faut recenser tous les modes de défaillances possibles (ou plutôt dont on souhaite la prise en compte) pour un système et définir au niveau du logiciel les modes de défaillance correspondants.

La correspondance mode de défaillance—données collectées sur le logiciel est plus ou moins difficile à effectuer selon la précision avec laquelle la collecte des données est (ou a été) effectuée. Cette correspondance est plus dure à effectuer si des techniques et des facilités de maintenance n'ont pas été prévues dès la conception. Elle dépend aussi en grande partie du cycle de vie où la défaillance a eu lieu :

- en développement, l'impact des défaillances sur le service, surtout au cours des premières phases n'est pas toujours perceptible et il faut souvent imaginer ce qui aurait pu se passer en vie opérationnelle car la conséquence sur le service délivré n'est pas observée, l'impact est plus facile à percevoir après la phase d'intégration,
- en vie opérationnelle, il est généralement plus facile de noter l'impact des défaillances sur le service délivré par le système puisque cet impact est effectivement observé.

IV-2-2-3 Lien taux de défaillance du logiciel — taux de défaillance selon les conséquences

Afin d'étudier le lien entre le taux de défaillance du logiciel et les différents taux de défaillance selon les modes de défaillance retenus, il faut distinguer le cas de la croissance de fiabilité du cas de la fiabilité stabilisée.

Fiabilité "croissante" :

Si les modes de défaillance sont choisis de telle sorte qu'ils soient mutuellement exclusifs (c'est-à-dire que chaque défaillance entraîne un seul type de conséquence), les processus de défaillance par mode peuvent ainsi être considérés comme indépendants. Dans ce cas le processus de défaillance global du logiciel résulte de la superposition de processus indépendants. Nous montrons dans l'annexe 3 que l'allure de l'évolution de l'intensité de défaillance est directement liée à l'allure de l'évolution de la somme des taux de défaillance des constituants.

En particulier :

- si les processus de défaillance selon les différents modes de défaillance sont tous à intensité strictement décroissante, le processus résultant est à intensité strictement décroissante,
- si un des processus de défaillance est à intensité non strictement décroissante mais globalement décroissante, la croissance locale se retrouve au niveau du processus global mais elle est atténuée (ou accentuée) par la décroissance (resp. croissance) de l'intensité due aux autres modes de défaillance,
- si on considère une politique de correction telle qu'une défaillance critique ou bloquante ou dont les conséquences sont jugées graves pour l'environnement du système donne lieu à des corrections pratiquement immédiates alors que les corrections relatives aux défaillances les moins graves peuvent être introduites uniquement au moment des changements de version

du logiciel (introduction de nouvelles spécifications), le taux de décroissance de l'intensité de défaillance entre deux versions successives est fonction uniquement du taux des défaillances critiques.

Fiabilité stabilisée :

Dans ce cas soit le logiciel est relancé sans correction après défaillance soit les corrections introduites n'entraînent pas de variation significative des taux de défaillance selon les différents modes. Ces processus de défaillance peuvent donc être considérés comme des processus de Poisson homogènes. Le processus de défaillance global résulte de la superposition de processus de Poisson homogènes, chacun d'eux correspondant à un mode de défaillance : c'est aussi un processus de Poisson homogène dont le taux est égal à la somme des taux de défaillance de ses constituants [Cox 70, Bar 75].
Les taux de défaillance relatifs aux modes de défaillances peuvent être évalués à l'aide du modèle hyperexponentiel.

A titre indicatif, pour l'étude que nous avons effectuée sur la fiabilité de l'autocommutateur téléphonique E-10, seules 13% des défaillances ont entraîné une indisponibilité totale du sous-système étudié et leur taux de défaillance en fiabilité stabilisée est de l'ordre de 5% du taux de défaillance de ce sous-système.

IV-2-3 Conclusions

Dans ce paragraphe nous avons montré l'intérêt d'appliquer les modèles de croissance de fiabilité par composant et selon les différents modes de défaillances.
Nous avons aussi montré que :
- en phase de croissance de fiabilité, le taux de défaillance à un instant donné est égal à la somme des taux de défaillance des différents constituants (composants ou conséquences) et l'évolution de l'intensité de défaillance est conditionnée par celle des constituants,
- en fiabilité stabilisée :
. si les composants sont indépendants vis-à-vis des défaillances, l'intensité de défaillance du logiciel est égale à la somme des intensités de défaillance de ses composants,
. si les conséquences des défaillances sont mutuellement exclusives, l'intensité de défaillance du logiciel est égale à la somme des intensités de défaillance par mode de défaillance.
- les intensités de défaillance relatives aux composants et aux différents modes de défaillances peuvent être évaluées par application du modèle hyperexponentiel aux données relatives à chacun des constituants.

IV-3- CALIBRAGE ET VALIDATION DES MODELES

A ce niveau, le ou les modèles de croissance de fiabilité qui vont être appliqués sont choisis ainsi que les mesures à évaluer et les donnés à fournir aux modèles pour évaluer ces mesures.
L'application d'un modèle de croissance de fiabilité est réalisée à l'aide :
- d'une **procédure d'inférence** permettant de **calibrer** le modèle, c'est-à-dire d'estimer les paramètres du modèle en fonction des données collectées sur le logiciel,
- d'une **procédure de prévision** qui consiste à remplacer, dans l'expression des mesures évaluées de la sûreté de fonctionnement, les paramètres par leurs valeurs estimées,
- d'un ou plusieurs **critères de validation** qui permettent de vérifier l'adéquation des résultats obtenus à la réalité (validation absolue par rapport à un critère) ou de comparer les valeurs de ces critères de validation relatifs à plusieurs modèles afin de choisir le "meilleur" (validation comparative).

La procédure de prévision ne posant aucun problème particulier, ne sera pas considérée dans ce qui suit. Les procédures d'inférence seront présentées de façon succincte dans le paragraphe suivant afin de fixer les idées. Les critères de validation seront ensuite analysés et discutés.

IV-3-1 Procédures d'inférence

La procédure d'inférence permet d'estimer les paramètres d'un modèle en fonction des données collectées (correspondant aux réalisations antérieures du processus). Une procédure d'inférence consiste à optimiser une expression à l'aide d'une procédure (appelé ici *procédure d'optimisation*).

L'expression à optimiser est fonction à la fois de la mesure à évaluer et de la méthode d'optimisation retenue : les deux méthodes les plus couramment utilisées sont la méthode des moindres carrés et la méthode du maximum de vraisemblance. Une fois l'expression établie, la procédure d'optimisation permet d'obtenir les valeurs des paramètres du modèle en fonction des données.

Le choix de la mesure à évaluer par les modèles de croissance de fiabilité impose le choix de la variable aléatoire qui sera privilégiée pour l'estimation des paramètres : si on a choisi d'évaluer un MTTF ou un taux de défaillance il faut optimiser selon T_i, si par contre on veut travailler sur le nombre cumulé de défaillances il faut utiliser la variable aléatoire N(t). Les paramètres obtenus pour optimiser l'estimation d'une mesure ne permettent pas

forcément d'obtenir de bonnes estimations pour les autres mesures. Ce choix n'a aucune influence sur l'adoption de la méthode d'optimisation.

D'un point de vue pratique, les deux méthodes d'optimisation (moindres carrés et maximum de vraisemblance) se ramènent à utiliser une **procédure d'optimisation** ; la même procédure peut être utilisée pour les deux méthodes. On peut utiliser l'une des procédures suivantes :

- procédure itérative de Newton-Raphson[Law 82],
- procédure de quasi Newton [Law 82],
- une procédure numérique de recherche directe d'optimum telle que celles de Powell [Pow 64] ou Fletcher et Reeves [Fle 64].

La première procédure nécessite l'établissement des expressions des dérivées premières et secondes par rapport à tous les paramètres alors que la seconde ne nécessite que les expressions des dérivées premières. Généralement, l'expression à optimiser est très complexe et un programme d'optimisation numérique est nécessaire. De tels programmes existent dans la plupart des bibliothèques classiques ; cependant pour tous ces programmes, l'utilisateur doit fournir des valeurs initiales pour les paramètres à estimer par la procédure d'optimisation et la convergence des algorithmes est plus ou moins rapide selon la valeur de ces paramètres. De telles procédures ne garantissent pas l'obtention de l'optimum absolu. Tous les résultats de l'évaluation reposent sur cette procédure. Il est donc primordial d'avoir un bon programme d'optimisation.

En résumé, pour calibrer un modèle il faut effectuer des choix à trois niveaux :

- choisir la variable aléatoire sur laquelle on va travailler en fonction des mesures à évaluer,
- choisir la méthode d'optimisation : méthode des moindres carrés ou méthode de maximum de vraisemblance,
- choisir la procédure d'optimisation afin de minimiser (moindres carrés) ou de maximiser (vraisemblance) l'expression établie à l'étape précédente.

De façon pratique, le calibrage d'un modèle est réalisé par un programme ; le premier choix est effectué par l'utilisateur selon les objectifs de son étude, les deux derniers doivent être transparents à l'utilisateur.

Le choix entre méthode des moindres carrés ou maximum de vraisemblance n'est pas évident et ne peut être effectué dans l'absolu : la comparaison détaillée à la fois sur les plans théorique et pratique effectuée dans [Sch 79] ne permet pas de trancher en faveur d'une méthode plutôt que l'autre.

IV-3-2 Critères de validation et de comparaison de modèles

Ces critères permettent d'apprécier "l'adéquation" des modèles de croissance de fiabilité aux données traitées. Ceci peut être fait de façon :
- *absolue* : on parle alors de valider un modèle ; le résultat étant que le modèle est à accepter ou à rejeter avec un certain niveau de confiance,
- *relative* : établir une échelle comparative entre les différents modèles utilisés.

L'adéquation d'un modèle peut être jugée selon deux aspects :
- **la capacité réplicative** du modèle qui traduit la capacité du modèle à reproduire le comportement passé du logiciel,
- **la capacité prédictive**, qui représente la capacité du modèle à prévoir le comportement futur du logiciel en fonction de son comportement passé.

Pour étudier la capacité réplicative on calibre le modèle avec toutes les données observées. Pour étudier la capacité prédictive on ne se sert que de la première partie des données observées pour calibrer le modèle et estimer les mesures de la sûreté de fonctionnement relatives à la deuxième partie, la capacité prédictive est alors évaluée sur cette dernière.
Que l'on s'intéresse à la capacité réplicative ou prédictive d'un modèle, les critères de validations sont les mêmes.

Trois critères sont présentés dans la suite : le critère de Kolmogorov-Smirnov, le critère de la vraisemblance préquentielle (appelé aussi approche préquentielle) et le critère des résidus.

IV-3-2-1 Critère de Kolmogorov-Smirnov

Soit $F_0(t)$ la fonction de répartition inconnue de la variable aléatoire intervalle entre défaillances et F(t) la fonction de répartition du modèle qu'on veut valider.

Le problème est de tester l'hypothèse : $F_0 = F$
contre l'hypothèse : $F_0 \neq F$

Les statistiques permettant de tester ces hypothèses se basent sur l'évaluation de la "distance" entre F(x) et la fonction de répartition empirique définie par :

$$F_n(t) = \frac{\text{nombre de } t_i \text{ inférieurs ou égaux à } t}{n}$$, pour tout t [Cox 66, Sap 78].

On utilise le théorème de la transformation intégrale pour tester si les points observés :

$$u_1 = F_1(t_1),\ u_2 = F_2(t_2),\ \ldots,\ u_n = F_n(t_n) \qquad \text{(IV-4)}$$

divisent l'intervalle unité de façon aléatoire, ou de manière équivalente s'ils sont uniformément distribués sur [0,1] .
Afin de tester cette hypothèse, on utilise la statistique de Kolmogorov-Smirnov (K-S) définie comme suit :

- les u_i sont classés selon l'ordre statistique :

$$u_{(1)} \leq u_{(2)} \leq .... \leq u_{(n)}$$

- la statistique D_n est donnée par :

$$D^+_n = \sup_{-\infty < x < +\infty} (F_n(x)-F(x)) = \max_{1 \leq i \leq n} \left(\frac{i}{n} - u_{(i)}\right),$$

$$D^-_n = \sup_{-\infty < x < +\infty} (F(x)-F_n(x)) = \max_{1 \leq i \leq n} \left(u_{(i)} - \frac{i-1}{n}\right), \qquad \text{(IV-5)}$$

$$D_n = \sup_{-\infty < x < +\infty} |F_n(x)-F(x)| = \max (D^+_n, D^-_n).$$

L'expression à gauche définit la statistique, celle de droite se prête mieux au calcul numérique.

Si $d_{n,\alpha}$ est telle que $P(D_n > d_{n,\alpha}) = \alpha$, alors le test statistique consiste à rejeter l'hypothèse (F_0=F) au niveau α (ou ce qui équivalent à accepter au niveau de confiance (1-α)) si la valeur observée de D_n est supérieure à $d_{n,\alpha}$. Il existe des tables qui donnent les valeurs limites $d_{n,\alpha}$ [Bra 68].

D_n mesure l'écart de F par rapport à F_n et permet ainsi de comparer différents modèles. Il faut préciser que les valeurs critiques sont calculées sous l'hypothèse que la distribution F à tester est entièrement connue : si des paramètres doivent être estimés, ces valeurs ne sont plus valables. Il faut alors considérer D_n comme purement indicatif et permettant de comparer divers modèles.

Remarques

1) Dans le cas où les modèles de croissance de fiabilité sont utilisés pour évaluer les mesures de la sûreté de fonctionnement comprises entre les indices k et m, ce critère est utilisé pour apprécier la qualité des modèles pour les indices k à m, n est alors égal à (m-k).

2) Un autre critère basé sur la distance de K-S a été développé dans [Kei 83] appelé le critère "y-plot" par analogie avec le critère de K-S qui est alors appelé "u-plot". Il part du principe que la distance de K-S classique détruit l'ordre d'arrivée des défaillances ; au lieu d'évaluer cette distance sur les u_i donnés par IV-4, on l'évalue sur les y_i définis par :

$$x_i = - \mathrm{Ln}(1-u_i)$$

$$y_i = \frac{\sum_{j=1}^{i} x_j}{\sum_{j=1}^{n} x_j} \qquad \text{(IV-6)}$$

Par définition les y_i sont dans l'ordre croissant. Dans (IV-5), il suffit de remplacer les $u_{(i)}$ par les y_i. La distance de K-S évaluée à partir des y_i peut être utilisée en complément de celle évaluée à partir des u_i, elle permet de tester la tendance des u_i.

IV-3-2-2 Approche préquentielle

L'approche préquentielle [Daw 84] ne peut être utilisée que de façon comparative : on définit un ou plusieurs systèmes de prévision séquentielle (SPS), les critères de comparaison permettent de mettre en évidence le système de prévision le mieux adapté aux données.

Système de prévision séquentielle

A partir des n premières réalisations de la variable aléatoire intervalle entre défaillances $(t_1, t_2, \ldots, t_n)$, on élabore une fonction de répartition de la prochaine observation de T_{n+1} ; soit F_{n+1} cette fonction.
Un système de prévision séquentielle est défini selon une loi qui, à chaque valeur de n et à chaque réalisation possible des T_i, associe un choix de F_{n+1}. On peut construire un SPS simplement en spécifiant une distribution conjointe des t_i et en prenant pour F_{n+1} la distribution conditionnelle de T_{n+1} connaissant les différentes réalisations des T_i .

Comparaison de systèmes de prévision

On considère un SPS noté P ; soit $f_n(t_n \mid t_1, t_2, \ldots, t_{n-1})$ la fonction densité de probabilité associée à la distribution $F_n(t_n \mid t_1, t_2, \ldots, t_{n-1})$. La fonction de vraisemblance préquentielle de P pour les données $(t_1, t_2, \ldots, t_n)$ est la fonction densité de probabilité conjointe des T_i définie par :

$$L_{P,n} = \prod_{i=1}^{n} f_i \; . \qquad \text{(IV-7)}$$

Soit Q un autre SPS et q_n la fonction densité de probabilité associée ; les deux SPS sont comparés grâce au rapport des vraisemblances préquentielles :

$$R_n = \frac{L_{Q,n}}{L_{P,n}}. \qquad \text{(IV-8)}$$

R_n converge vers une limite finie R avec une probabilité 1. Si la séquence des réalisations R_n tend vers l'infini quand n tend vers l'infini (P<< Q), on se trouve ainsi en présence d'un événement qui ne peut être régi par P, et on peut ainsi discréditer P au profit de Q.

Si P<< Q on ne peut que discréditer P au profit de Q, et on ne peut pas en déduire par exemple que Q est un bon SPS dans ces conditions ; Q est meilleur que P.

Si R_n ne tend ni vers zéro ni vers l'infini, P et Q sont alors considérés comme deux systèmes équivalents : P ~ Q, et on ne peut choisir définitivement entre les deux systèmes. Ceci est le principal inconvénient de ce critère car, souvent pour des cas pratiques, on se trouve dans cette situation où on ne peut conclure. Il peut cependant être utilisé comme critère de comparaison [Abd 86].

IV-3-2-3 Critère des résidus

Ce critère est basé sur le calcul de l'écart entre la valeur observée et la valeur estimée de la variable aléatoire T_i, intervalle de temps entre la (i-1) ième et la i ième défaillances, ou N(j) le nombre cumulé de défaillances jusqu'à la j ième unité de temps.

Cas où la variable aléatoire est T_i.

On appelle résidu l'observation de la variable aléatoire différence entre la réalisation et l'estimation par le modèle :

$$R_i = T_i - MTTF_i, \qquad i = k, \ldots, m. \qquad \text{(IV-9)}$$

avec :

- $MTTF_i$: i ième intervalle de temps estimé par le modèle,
- k : indice du premier intervalle de temps estimé par le modèle,
- m : indice du dernier intervalle de temps estimé par le modèle.

R_i est de moyenne nulle et de variance égale à celle de T.

La réalisation de cette variable aléatoire est notée : $r_i = t_i - MTTF_i$.

Dans la pratique, pour un modèle donné, on évalue soit :

- la somme des résidus : $r_s = \sum_{i=k}^{m} r_i$, (IV-10)

- la somme des carrés des résidus : $r_c = \sum_{i=k}^{m} r_i^2$, (IV-11)

- la somme des valeurs absolues des résidus : $r_a = \sum_{i=k}^{m} |r_i|$, (IV-12)

- la somme des valeurs absolues des résidus relatifs : $r_r = \sum_{i=k}^{m} \left|\frac{r_i}{t_i}\right|$. (IV-13)

Si on veut comparer deux modèles A et B ayant respectivement pour résidus r_{xA} et r_{xB}, selon la formulation choisie ($x \in \{s,c,a,r\}$), le meilleur modèle correspond à celui qui a le résidu le plus faible.

La comparaison des modèles ne donne pas le même résultat selon la formulation choisie. L'utilisateur opte pour la formulation qui répond le mieux à ses besoins :
- la somme des résidus r_s permet de voir si l'estimation est biaisée ainsi que le sens du biais : estimation globalement optimiste ou pessimiste,
- r_c privilégie les modèles qui s'écartent faiblement des observations et défavorisent ceux dont certains points s'écartent considérablement de la réalisation même si tous les autres points collent tout à fait aux données observées,
- r_a traite tous les écarts de la même façon,
- r_r donne plus d'importance aux premières données en présence de croissance de fiabilité car les intervalles entre défaillances sont plus faibles au début qu'à la fin et est mal adapté à un ensemble de données comprenant un ou plusieurs intervalles nuls.

La comparaison de modèles est effectuée au travers de la comparaison de leurs résidus, généralement on se sert soit de la moyenne :

$$\frac{r_x}{m-k}\,, \qquad \text{(IV-14)}$$

soit de la valeur normée par rapport au temps total d'observation :

$$\frac{r_x}{s_n}\,, \qquad \text{(IV-15)}$$

avec : $s_n = \sum_{i=k}^{m} t_i$.

Pour notre part, nous évaluons r_s ce qui permet d'avoir une indication sur le biais ainsi que l'ordre de grandeur de l'écart "relatif" entre le MTTF observé et estimé. Afin de faire apparaître les bornes de la sommation, ce résidu est noté :

$$R_{k,m} = \frac{\sum_{i=k}^{m}(t_i - MTTF_i)}{\sum_{i=k}^{m} t_i} \qquad \text{(IV-16)}$$

Dans le cas où r_s est très proche de zéro, il faut évaluer la somme des carrés des résidus qui constitue un critère de la qualité d'adéquation des modèles [Fon 85].
En fait, $R_{k,m}$ ne peut servir qu'à titre indicatif, l'étude de l'évolution du résidu au cours du temps est beaucoup plus riche en renseignements, et il est conseillé de faire apparaître, autant que possible, les valeurs successives des résidus.

Remarques

1) Indépendamment de la formulation retenue, l'utilisation de la somme des résidus (sous l'une des quatre formes précédentes) est très pratique car elle permet de manipuler une valeur par modèle ; cependant la sommation peut masquer le sens de variation locale de l'écart. Il ne faut donc pas perdre de vue les valeurs successives des résidus, elles pourraient révéler certains défauts des modèles : inaptitude à suivre une forte variation de tendance, ou influence d'une donnée douteuse qui entraîne à elle seule un résidu élevé...

2) Bien qu'à notre connaissance, le résidu ne soit pas utilisé couramment pour vérifier l'adéquation des modèles de croissance de fiabilité du logiciel, il nous semble un très bon indicateur de l'écart entre l'intervalle entre défaillances estimé et celui qui est réellement observé. De plus, contrairement aux autres critères, il est directement lié à la grandeur physique observée. En fait le résidu est plus couramment utilisé dans le cas où l'on s'intéresse au nombre cumulé de défaillances et dans ce cas il est appelé erreur.

Cas où la variable aléatoire est N(t).

Le résidu est défini par :

$$r_j = N(i) - H(i) \qquad j = l, \ldots, p. \qquad \text{(IV-17)}$$

avec :

- N(i) : le nombre de défaillances jusqu'à la fin de la i ième unité de temps,
- H(i) : valeur de la fonction moyenne à la fin de la i ième unité de temps,
- l : unité de temps à partir de laquelle commence l'estimation de N(t),

- p : dernière unité de temps de l'estimation de N(t).

Toutes les formulations données plus haut sont également valables dans ce cas :

- la somme des résidus : $r_s = \sum_{i=1}^{p} r_i$, (IV-18)

- la somme des carrés des résidus : $r_c = \sum_{i=1}^{p} r_i^2$, (IV-19)

- la somme des valeurs absolues des résidus : $r_a = \sum_{i=1}^{p} |r_i|$, (IV-20)

- la somme des valeurs absolues des résidus relatifs : $r_r = \sum_{i=1}^{p} \left| \frac{r_i}{n(i)} \right|$ (IV-21)

où n(i) est le nombre de défaillances durant la i ième unité de temps.

Quand on considère la variable aléatoire intervalle entre défaillances, sa distribution peut être approchée par une distribution exponentielle pour laquelle la moyenne est égale à l'écart type ; la somme des résidus constitue un bon critère d'adéquation. Dans le cas où la variable aléatoire est le nombre de défaillances on cherche un modèle qui estime de la façon la plus précise la valeur du nombre cumulé de défaillances, il faut prendre soit la somme des valeurs absolues soit la somme des carrés des résidus. Cette somme peut être normée par le nombre d'intervalles de temps considérés. Pour notre part nous avons évalué :

$$E_{1,p} = \frac{\sum_{i=1}^{p} |N(i) - H(i)|}{p-1} \qquad \text{(IV-22)}$$

Comme pour le cas précédent, nous conseillons de faire apparaître les différentes valeurs des résidus dont l'évolution est beaucoup plus significative que $E_{1,p}$.

IV-3-3 Conclusions

Dans ce paragraphe nous avons présenté brièvement :
- les principes des méthodes de calibrage des modèles de croissance de fiabilité (estimation des paramètres à l'aide des données collectées sur le logiciel),
- quelques critères de validation et surtout de comparaison de ces modèles.

En ce qui concerne le calibrage des modèles, pour les modèles que nous avons utilisés (et pour lesquels nous avons écrit les programmes d'évaluation des mesures de la sûreté de

fonctionnement), nous avons choisi la méthode de maximum de vraisemblance pour estimer les paramètres. En ce qui concerne la procédure d'optimisation :
- pour les modèles à trois paramètres, nous nous sommes servie d'une bibliothèque existante (Harwell), le programme étant basé sur la procédure d'optimisation directe de M.J.D.Powell,
- pour les modèles à deux paramètres, nous avons utilisé un programme d'optimisation basé sur la méthode de Newton Raphson.

Trois critères permettant d'apprécier la qualité d'un modèle de croissance de fiabilité de logiciel ont été présentés ; nous avons vu que ces critères ne peuvent pas être utilisés dans l'absolu mais de façon relative.

Le critère de Kolmogorov-Smirnov est, à l'origine, un critère de validation de modèles associés à des processus de renouvellement stricts. Dans le cas des processus de renouvellement généralisés, il peut être utilisé pour comparer des modèles.

Le critère de la vraisemblance préquentielle n'est valable que dans le cas où un modèle est dominant par rapport aux autres, ce qui n'est pas le cas le plus fréquent.

Le critère des résidus est lui aussi défini de façon comparative ; il permet cependant d'avoir un ordre de grandeur de "l'erreur" entre l'estimation et l'observation de l'intervalle entre défaillances (ou du nombre cumulé de défaillances).

Notre préférence va vers ce dernier critère :

- dans le cas où la variable aléatoire est l'intervalle entre défaillances on évaluera le résidu normé par la durée de l'estimation,
- dans le cas où la variable aléatoire est le nombre de défaillances on évaluera un résidu moyen (somme des valeurs absolues divisée par le nombre de valeurs estimées).

La distance de K-S sera évaluée dans le cas où la variable aléatoire est l'intervalle entre défaillances.

IV-4- CONCLUSIONS

Ce chapitre nous a permis de déterminer les mesures les plus significatives à évaluer par les modèles de croissance de fiabilité ; deux mesures sont intéressantes à évaluer : le MTTF et le taux de défaillance, le taux d'occurrence des défaillance (ou intensité) se déduisant de cette dernière. Nous avons aussi vu comment le suivi de l'évolution du nombre cumulé de défaillances en développement peut constituer un moyen d'aide à la vérification du

bon déroulement du développement. Enfin l'estimation du nombre de défaillances pour la période future peut servir à la planification du développement et de la maintenance.

Nous avons montré ensuite l'intérêt de l'application des modèles de croissance de fiabilité (notamment le modèle hyperexponentiel) par composant et par conséquence pour l'évaluation de la fiabilité du logiciel en vie opérationnelle.

Nous avons ensuite présenté brièvement quelques méthodes de calibrage de modèles de croissance de fiabilité ainsi que des critères de validation et surtout de comparaison de modèles ; deux critères ont été retenus : le critère de Kolmogorov-Smirnov et le critère des résidus qui peut avoir plusieurs formulations selon ce que l'utilisateur cherche à faire apparaître.

L'ensemble de la méthode proposée pour l'évaluation de la fiabilité d'un logiciel sera appliqué (dans le prochain et dernier chapitre) à deux systèmes réels qui sont deux autocommutateurs téléphoniques.

CHAPITRE V

ETUDE DE LA FIABILITE DU LOGICIEL

DE DEUX AUTOCOMMUTATEURS TELEPHONIQUES

INTRODUCTION

Les précédents chapitres ont permis de présenter en détail les différentes aspects de la méthode globale que nous proposons pour l'évaluation de la sûreté de fonctionnement du logiciel. Dans ce chapitre nous appliquons cette méthode aux logiciels de deux autocommutateurs téléphoniques de caractéristiques différentes afin d'étayer certains aspects de cette méthode et de montrer le type et l'intérêt des résultats qu'elle permet d'obtenir.

Cette méthode revêt deux étapes : l'étape système et l'étape appréciation et évaluation de la fiabilité. Ces deux étapes n'étant pas totalement indépendantes, elles ne seront pas totalement dissociées dans notre présentation : nous donnons juste une présentation générale du système, certaines informations sont fournies au moment où elles interviennent pour expliquer un phénomène particulier. Le plan adopté pour l'étude de chaque système est le suivant :

- présentation succincte du système et des données collectées,
- prétraitement des données,
- application des modèles de croissance de fiabilité.

Les deux études ont été orientées de façon différente afin d'illustrer des aspects complémentaires :

- L'étude de l'autocommutateur d'ALCATEL-CIT permet de montrer l'amélioration de l'évaluation de la fiabilité due à l'application des modèles par palier. Nous évaluerons aussi la fiabilité en vie opérationnelle selon les modes de défaillance ainsi que la fiabilité des différents composants du logiciel, en vie opérationnelle.

- L'étude du TROPICO-R sert d'exemple d'utilisation simultanée de deux types de modèles de croissance de fiabilité : afin d'estimer le nombre cumulé de défaillances d'une part et d'évaluer la fiabilité en vie opérationnelle d'autre part.

Quatre modèles de croissance de fiabilité de logiciel de nature différente ont été sélectionnés :

- le modèle de Musa [Mus 75], noté MU,
- le modèle de Littlewood-Verral [Lit 73], noté LV,
- le modèle hyperexponentiel, noté HE,
- un modèle en forme de S : delayed S-Shaped model [Yam 823], noté SS.

Ce choix permet de couvrir les divers aspects mis en évidence dans les chapitres précédents :

- croissance monotone de fiabilité avec un taux tendant vers zéro quand t tend vers l'infini, le pas de décroissance étant constant (**MU**) ou lui même décroissant (**LV**),
- croissance monotone avec un taux de défaillance tendant vers une constante non nulle (**HE**),
- taux de défaillance croissant puis décroissant (**SS**).

Les trois premiers modèles sont bien adaptés au cas où la variable aléatoire considérée est l'intervalle entre défaillances, alors que le dernier est plus adapté au nombre cumulé de défaillances.

V-1- AUTOCOMMUTATEUR TELEPHONIQUE E-10

Il s'agit d'un sous-ensemble de l'autocommutateur téléphonique E-10 développé en France par ALCATEL-CIT sur lequel ont été collectées des données de fiabilité du logiciel. L'étude complète relative à la fiabilité de ce logiciel est effectuée dans [Sab 87]. Ce système a été observé sur une période de trois ans incluant une partie du développement et une partie de la vie opérationnelle. Durant cette période il y a eu deux changements de spécifications : les données disponibles correspondent donc à trois versions successives.

Le matériel est tolérant aux fautes : il est composé d'une unité active (pilote) et d'une réserve ; le basculement de l'unité pilote vers l'unité réserve est effectué toutes les 24 heures afin de réduire la latence des fautes. La gestion de la tolérance aux fautes du matériel est assurée par un composant logiciel appelé DEFENSE.

Le nombre de systèmes en vie opérationnelle n'a cessé de croître tout au long de la période considérée pour atteindre à peu près 1400. Au moment où l'étude de fiabilité a commencé, la collecte de données était déjà effectuée. Les informations disponibles sont de deux types : celles relatives aux défaillances et celles relatives aux corrections. L'organisation au sein de l'entreprise est telle que, selon la phase du cycle de vie du logiciel où la défaillance a eu lieu, une fiche de défaillance doit être rédigée ou non, alors que toutes les corrections donnent lieu à la rédaction d'une fiche de correction. Ceci explique le fait que nous disposons de 58 fiches de défaillances contre 136 fiches de corrections et que souvent il a été impossible de faire une

correspondance entre les éléments des deux ensembles. De ce fait nous avons décidé de travailler sur ces deux ensembles de façon séparée en faisant les rapprochements entre les deux, chaque fois que cela était possible.
Le logiciel occupe un volume d'environ 100 Kilo octets et est écrit en Assembleur.

V-1-1 Prétraitement des données

V-1-1-1 *Recherche des données douteuses*

Le test des données douteuses appliqué aux 58 données de défaillance détecte trois données trop élevées qui correspondent respectivement aux 58, 53 et 56 ièmes valeurs ; ceci peut dénoter une forte croissance de fiabilité en fin de période. La croissance de fiabilité est encore plus visible sur les 136 données relatives aux corrections : 17 données ont été détectées trop élevées et pratiquement toutes ces données sont situées au delà de l'indice 100.
Aucune donnée douteuse ne sera écartée par la suite, car comme nous l'avons signalé dans la démarche proposée, elles sont à l'origine (ou la conséquence) de la croissance de fiabilité observée sur les données.

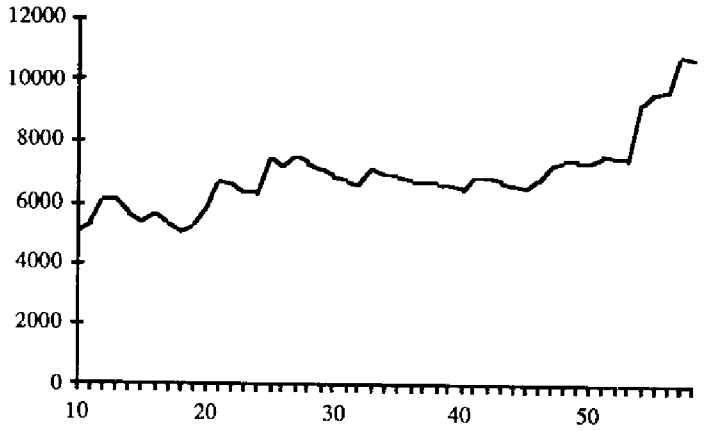
V-1-1-2 *Tests de tendance*

Les trois tests graphiques ont été effectués pour les données de défaillance et tous les trois montrent une fiabilité légèrement décroissante (plutôt stabilisée) au début de la période d'observation. Ceci est illustré par les courbes de la figure V-1 a, b et c qui donnent respectivement l'évolution de la moyenne arithmétique en fonction du numéro de la défaillance, le nombre cumulé de défaillances et le nombre de défaillances par périodes de trois mois (ROCOF). Bien que la courbe du ROCOF ait la forme d'une cloche, le changement de courbure au niveau de N(t) (courbe b) n'est pas très prononcé.
Le test de Laplace montre aussi que, malgré une croissance de fiabilité globale, le logiciel passe par des périodes de fluctuation (figure V-2).

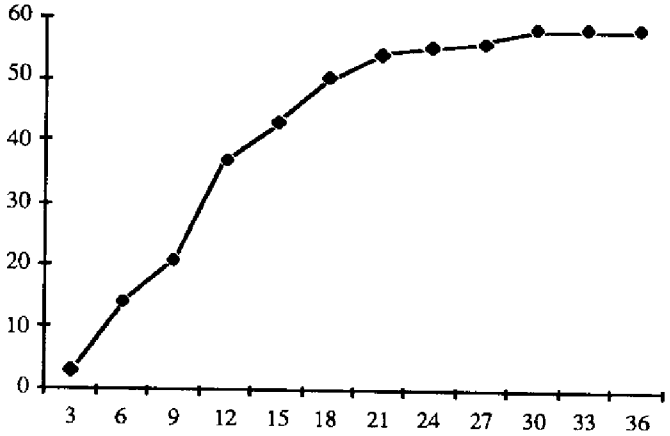
En ce qui concerne les données de correction, il y a manifestement croissance de fiabilité de façon régulière avec une forte croissance vers la fin. Nous donnons à la figure V-3 les courbes de la moyenne arithmétique et du nombre cumulé de défaillances et le coefficient de tendance de Laplace (qui est presque toujours négatif).

A titre indicatif, la répartition du nombre de défaillances et de corrections sur les trois ans est la suivante :

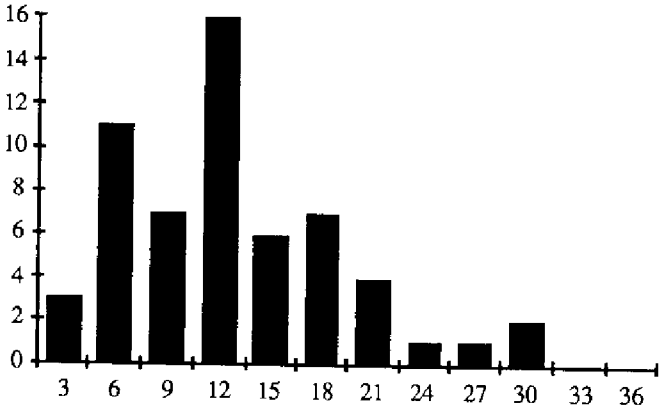
- 37 défaillances ont été notées la première année pour 113 corrections,
- 18 défaillances et 18 corrections la deuxième année,
- 3 défaillances et 5 corrections la dernière année.



-a- Moyenne arithmétique en fonction du numéro de défaillance



-b- Nombre cumulé de défaillances en fonction du temps (en mois)



-c- Nombre de défaillances par période de trois mois

Figure V-1 : Tests de tendance pour les données de défaillance

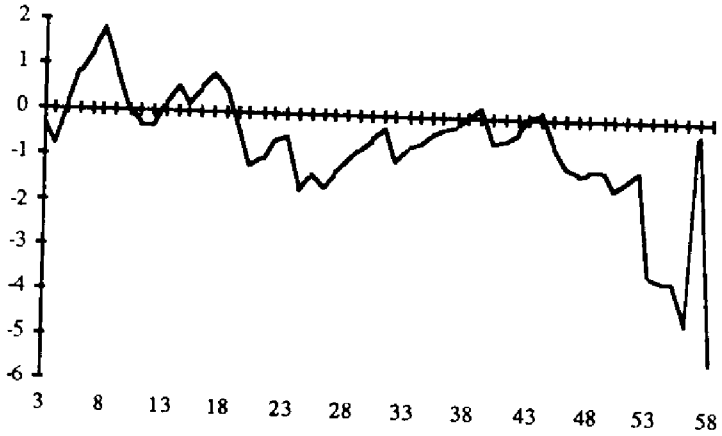


Figure V-2: Test de Laplace pour les données de défaillance

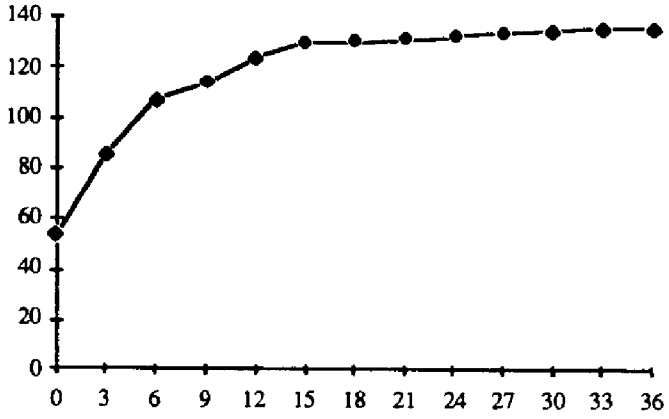
On peut remarquer que :

- la fréquence des corrections (ainsi que celle des défaillances qui ont été notées) a considérablement baissé,
- il y a presque correspondance entre défaillances et corrections les deux dernières années (vingt dernières données).

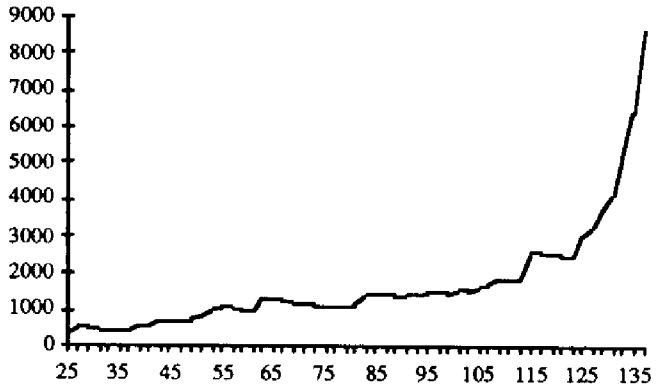
V-1-1-3 *Conclusions*

Il est intéressant de constater que, malgré les changements de versions il y a eu croissance de fiabilité globale, ce qui confirme la proposition faite au premier chapitre : la décroissance due au changement de version est compensée par les corrections introduites en même temps. Il y a eu cependant des fluctuations locales de la fiabilité dues à la fois aux changements de spécifications et des conditions d'utilisation. Mais nous avons indiqué que toutes les défaillances ne sont pas notées ; nous ne pouvons donc pas tirer de conclusions à leur sujet. Nous savons cependant que la période de légère décroissance a été entraînée par des essais sur site en parallèle avec le trafic normal : les conditions d'utilisation ayant changé (augmentation de la charge du système), une décroissance apparente de la fiabilité a été perçue.

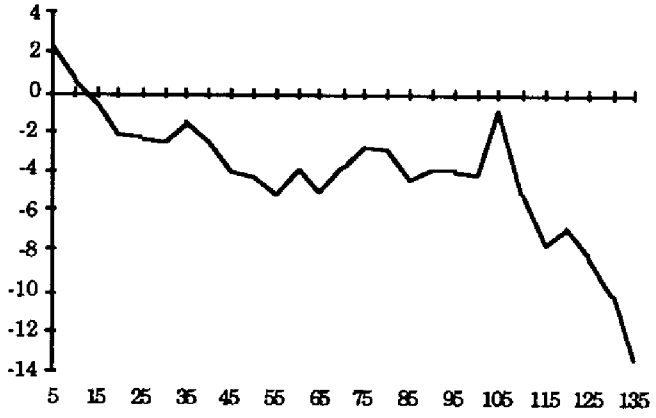
Nous nous servirons du résultat des tests de tendance pour l'application des modèles de croissance fiabilité au paragraphe suivant. En ce qui concerne les données douteuses, nous avons vu que, bien que le test correspondant indique l'existence d'un grand nombre, aucune donnée ne sera écartée car elles sont toutes situées en fin de période d'observation dénotant ainsi une forte croissance de fiabilité en vie opérationnelle.



-a- Nombre cumulé de corrections en fonction du temps (en mois)



-b- Moyenne arithmétique en fonction du numéro de défaillance



-c- Coefficient de Laplace en fonction du numéro de défaillance

Figure V-3 : Tests de tendance pour les données de correction

V-1-2 Application des modèles

Les modèles de croissance de fiabilité seront d'abord appliqués sur toutes les données, les résultats des tests de tendance nous permettront d'améliorer les estimations en appliquant les modèles par palier. Les données seront ensuite regroupées par conséquence (resp. par composant) afin d'évaluer la contribution de chaque mode de défaillance (resp. composant) sur la fiabilité du logiciel en vie opérationnelle.

V-1-2-1 Utilisation de toutes les données

Deux ensembles de données sont disponibles : celles concernant les défaillances et celles concernant les corrections. Ces dernières, ne présentant que de la croissance de fiabilité, ne seront pas traitées dans ce qui suit.

Les résultats de l'application des trois modèles MU, LV et HE aux données de défaillances sont rassemblés dans les figures V-4 et V-5 qui donnent respectivement l'évolution des MTTF et des taux de défaillance estimés par les modèles.
La première colonne du tableau de la figure V-4 indique le numéro de la défaillance, la deuxième donne l'intervalle entre défaillances observé, les troisième, quatrième et cinquième colonnes donnent respectivement les résultats des estimations du $MTTF_i$ par les modèles MU, LV, HE. Les données 1 à 8 ont servi à calibrer le modèle au départ pour estimer le taux de la prochaine défaillance, les estimations sont faites pas à pas, c'est-à-dire que pour évaluer le taux d'occurrence de la (i+1) ième défaillance on calibre à nouveau le modèle à l'aide des données 1 à i.
On note ensuite les valeurs des résidus et des distances de Kolmogorov-Smirnov pour les modèles LV et HE. Pour le modèle MU, seul un résidu partiel a pu être calculé sur les données où le modèle était applicable. Ce résidu ne pourra pas être comparé aux autres. En effet le modèle MU n'est applicable que dans le cas où la fonction de vraisemblance possède un optimum, ce qui correspond aussi au cas où le coefficient de tendance de Laplace est négatif (croissance de fiabilité). De façon plus générale, le modèle MU réagit très rapidement à une croissance (ou décroissance) dans les valeurs observées ; ceci l'amène à estimer parfois des taux de défaillance nuls : il ne supporte pas une décroissance de fiabilité, même passagère, dans les observations.
On observe une longue période durant laquelle il n'y a pratiquement pas croissance de fiabilité. A l'inverse du modèle MU, les modèles LV et HE montrent leur faculté à suivre une constance ou une décroissance de fiabilité. On peut remarquer que le modèle LV donne des résultats (résidus et distances de Kolmogorov-Smirnov) plus faibles que le modèle HE et donc

i	t_i	$MTTF_i$, MU	$MTTF_i$, LV	$MTTF_i$, HE
9	8818.02	++++++	127.97	5093.69
10	7596.76	++++++	7174.59	5560.49
11	13525.53	++++++	9352.10	5786.84
12	6497.44	7427.00	12228.24	6561.49
13	723.06	7183.63	11371.89	6555.89
14	724.18	++++++	9766.56	6069.26
15	9603.34	++++++	7793.62	5657.71
16	739.84	++++++	9119.80	5940.47
17	740.96	++++++	7457.04	5592.84
18	9052.57	++++++	4798.90	5289.30
19	17137.82	++++++	7312.63	5511.14
20	21848.32	7354.32	9835.60	6156.98
21	4073.95	11566.99	12104.40	6983.04
22	815.91	9946.09	11460.17	6837.54
23	4935.73	8246.75	10701.88	6549.75
24	33783.77	8341.63	10718.20	6477.48
25	867.37	14715.06	13561.50	7665.65
26	16883.87	13038.93	13026.63	7382.07
27	889.74	14090.78	13692.12	7762.43
28	2679.30	11818.92	13204.07	7497.56
29	894.22	10322.62	12517.36	7318.55
30	2692.72	8930.64	11722.16	7089.63
31	898.69	8050.36	10756.24	6937.52
32	24122.26	7226.80	9304.13	6736.65
33	928.90	9742.44	12391.61	7297.73
34	5613.66	8655.83	11392.40	7098.78
35	1875.69	8298.25	10934.07	7053.00
36	4717.20	7561.95	9865.96	6900.70
37	4745.17	7259.55	9821.29	6838.65
38	2857.17	6991.67	9150.77	6780.49
39	953.51	++++++	8495.16	6674.31
40	26560.20	++++++	7335.21	6524.27
41	4946.54	++++++	10207.47	7037.84
42	990.43	7870.14	10205.14	6985.41
43	1985.33	7202.04	8991.30	6838.80
44	993.78	++++++	8240.81	6723.41
45	26594.58	++++++	7178.97	6589.99
46	23044.56	8303.83	9832.50	7045.02
47	12730.85	10027.36	11572.54	7401.09
48	5332.49	10415.78	12554.20	7516.87
49	7520.30	10054.85	11606.62	7469.50
50	21934.06	10159.09	11483.56	7469.50
51	3300.18	11410.98	12742.53	7765.83
52	1101.18	10684.96	12024.99	7677.03
53	108167.45	11426.19	11547.17	7547.73
54	25732.42	26662.63	16615.08	9483.10
55	16118.65	28222.96	17529.96	9790.70
56	71578.37	33967.49	18281.62	9907.27
57	3914.34	40851.40	20697.44	11028.91
58	111542.16	52754.25	20244.39	10901.43
RESIDU ($R_{9,58}$) =		+1.61E-01	1.98E-01	4.82E-01
DISTANCE DE KOLMOGOROV-SMIRNOV =			1.80E-01	1.87E-01

++++++ Modèle non applicable (u > 0)

Figure V-4 : Application des modèles aux données de défaillance (en jours)

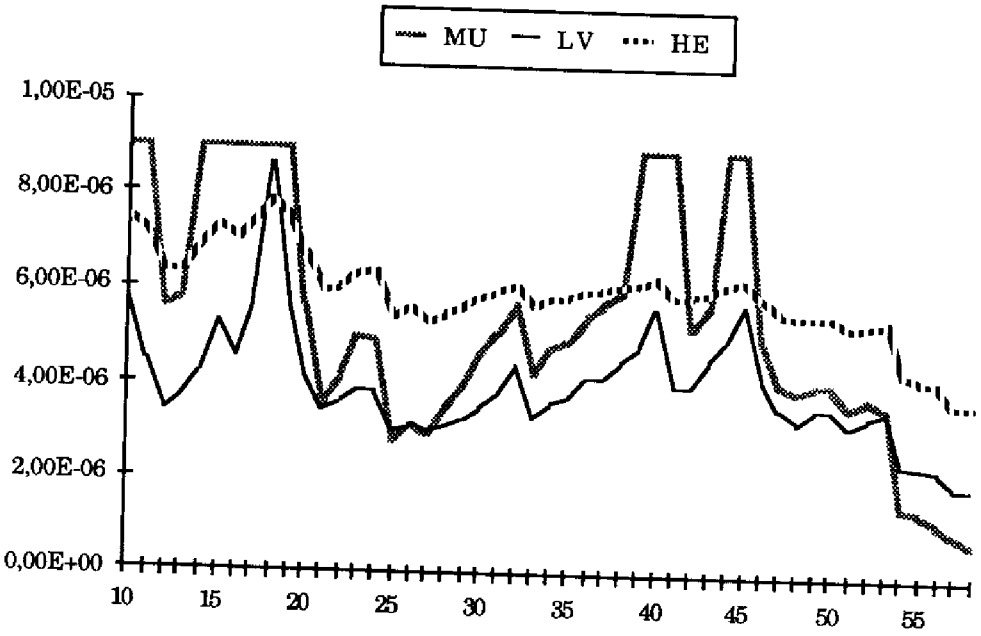


Figure V-5 : Taux de défaillance (h^{-1}) pour les données de défaillance

s'adapte mieux aux données étudiées. En contrepartie le modèle LV suit "trop" bien les données et ne réalise pas le lissage nécessaire, que donne le modèle HE.

Ce lissage permet d'avoir une valeur plus stable du MTTF estimé et de ne pas tenir compte des fluctuations (souvent microscopiques et accidentelles) au cours du temps, d''autant plus que ces fluctuations sont pour la plupart dues à des périodes de ralentissements ponctuels de l'activité de validation. Le modèle HE éprouve de grandes difficultés à suivre une forte croissance de fiabilité (décroissance du taux) survenant brusquement. Ceci s'explique par le fait que le modèle est beaucoup plus sensible aux premières données. Enfin les résultats sont relativement bons car le critère de Kolmogorov-Smirnov indique que les ajustements sont significatifs à 5%.

V-1-2-2 Application des modèles par palier

La forte croissance de fiabilité observée sur les données collectées correspond sensiblement au passage de la phase de validation à la vie opérationnelle. La discontinuité que l'on rencontre pose des problèmes d'applicabilité des modèles. Aussi, une élimination progressive des premières données a été réalisée. Ceci nous a permis de diminuer progressivement l'influence restrictive et pénalisante des données de début de validation. Pour ce faire, les modèles LV et HE ont été appliqués par palier.

Sept paliers ont été définis sur la période d'observation, notés P_1 à P_7, et l'ensemble des données a été réparti selon ces paliers. Ces paliers correspondent à des paliers techniques : les frontières coïncident avec des moments de changement de phase ou à l'introduction d'une nouvelle version ou à la mise en service de nouvelles instances.

Les résultats de l'application des modèles LV et HE par palier (de P_2 à P_7) sur les données de défaillance sont rassemblés dans les figures V-6 et V-7. En comparant avec les figures V-4 et V-5 on remarque une nette amélioration des résultats. Les estimations s'ajustent beaucoup mieux aux observations et les résidus sont nettement plus faibles : de 4 fois pour le modèle LV à 10 fois plus faibles pour le modèle HE. Les MTTF de fin de période d'observation sont plus significatifs et donnent une meilleure évaluation de la fiabilité en vie opérationnelle. Pour ce cas aussi, LV est plus sensible aux fluctuations locales.

Il est clair que le choix du nombre et de la taille des paliers a une influence prépondérante sur les résultats obtenus. L'étude de l'influence du nombre moyen de données par palier effectuée dans [Sab 87] montre que le meilleur résidu est obtenu pour des paliers de longueur moyenne égale à 2 alors que la meilleure distance de Kolmogorov-Smirnov est obtenue pour des paliers de longueur égale à 16, d'où la nécessité d'un compromis entre ces deux critères.

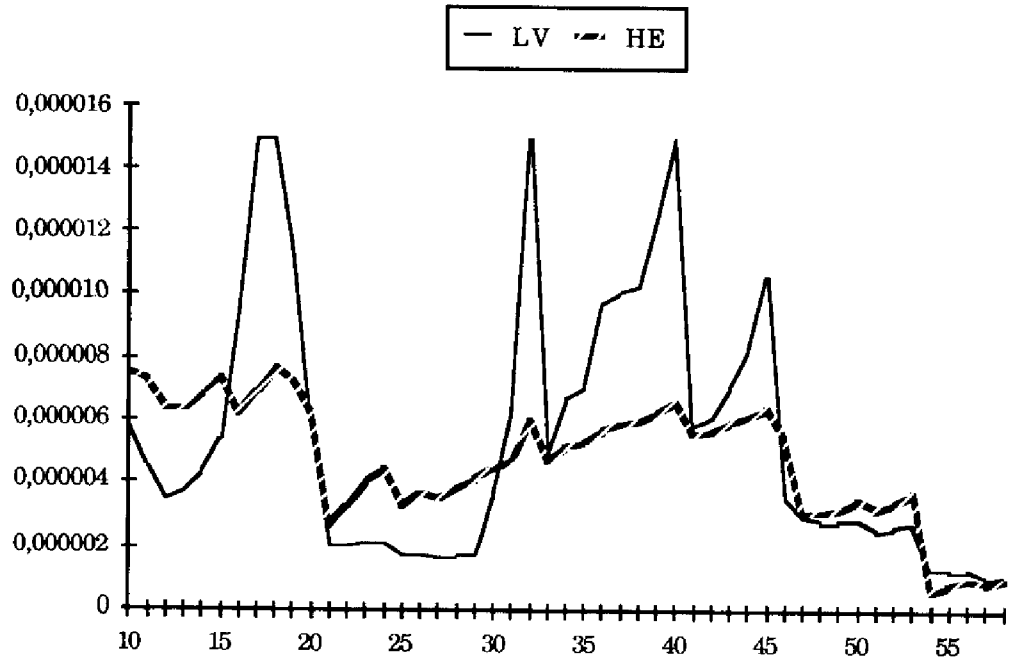


Figure V-7 : Taux de défaillance (h^{-1}) par palier pour les données de défaillance.

	i	t_i	$MTTF_i$, LV	$MTTF_i$, HE
	9	8818.02	127.97	5093.69
	10	7596.76	7174.59	5560.49
	11	13525.53	9352.10	5786.84
P2	12	6497.44	12228.24	6561.49
	13	723.06	11371.89	6555.89
	14	724.18	9766.56	6069.26
	15	9603.34	7793.62	5657.71
	16	739.84	4336.35	6783.88
	17	740.96	980.46	6027.04
P3	18	9052.57	-454.53	5440.13
	19	17137.82	3644.75	5804.15
	20	21848.32	7019.60	6832.69
	21	4073.95	20396.10	15757.30
	22	815.91	20202.01	12834.38
	23	4935.73	19341.80	10429.85
	24	33783.77	19339.88	9512.19
	25	867.37	23962.86	12981.95
P4	26	16883.87	23318.53	11465.43
	27	889.74	24752.69	12061.06
	28	2679.30	24063.18	10950.51
	29	894.22	23367.10	10199.70
	30	2692.72	11595.84	9423.68
	31	898.69	6771.40	8904.49
	32	24122.26	-226.26	6971.09
	33	928.90	8427.79	8918.54
	34	5613.66	6238.23	8107.87
	35	1875.69	5890.38	7877.37
	36	4717.20	4283.60	7369.09
	37	4745.17	4135.52	7160.50
P5	38	2857.17	4052.72	6983.96
	39	953.51	3345.25	6701.53
	40	26560.28	1881.63	6329.62
	41	4946.54	7204.59	7546.90
	42	990.43	6864.39	7400.70
	43	1985.33	5875.08	7058.49
	44	993.78	5050.47	6799.99
	45	26594.58	3942.07	6517.27
	46	23044.56	11838.26	7706.04
	47	12730.85	14267.91	13564.42
	48	5332.49	14971.58	13713.73
P6	49	7520.30	14677.60	13141.19
	50	21934.06	14621.21	12132.80
	51	3300.18	16177.02	13104.35
	52	1101.18	15625.57	12256.11
	53	108167.45	14769.33	11240.90
	54	25732.42	31707.91	66374.81
	55	16118.65	32307.10	51221.37
P7	56	71578.37	30812.71	42364.31
	57	3914.34	39303.74	48314.99
	58	111542.16	35627.80	40709.49
RESIDU ($R_{9,58}$) =			5.81E-02	5.83E-02

Figure V-6 : Application des modèles LV et HE par palier (Temps en jours)

V-1-2-3 Application des modèles par conséquence

Les conséquences des défaillances peuvent être extraites des fiches de défaillance. Elles ont été réparties en quatre classes correspondant aux conséquences les plus fréquemment rencontrées :

A : **indisponibilité générale** du sous-système,

B : **indisponibilité partielle** : le service offert aux utilisateurs est dégradé (temps d'attente d'établissement de communications plus longs, capacité de traitement d'appels réduite, ruptures de quelques communications téléphoniques),

C : **abandon d'un traitement d'exploitation,**

D : **perte d'une unité** matérielle : comme le système est redondant les fonctions du système ne sont pas affectées.

La répartition du nombre de défaillances par conséquence est la suivante :
- 7 défaillances ont entraîné une indisponibilité générale,
- 17, une indisponibilité partielle,
- 18, l'abandon d'un traitement d'exploitation,
- 16, la perte d'une unité.

Les résultats d'application du modèle hyperexponentiel pour évaluer le taux de défaillance résiduel (λ_S) en vie opérationnelle sont résumés dans la figure V-8 qui donne aussi la valeur des résidus (RES, dont l'expression est donnée par IV-16) ainsi que la distance de Kolmogorov-Smirnov (DKS).

Mode de défaillance	$\lambda_S(10^{-7}\ h^{-1})$	RES	DKS
Indisponibilité générale	1.21	0.16	0.53
Indisponibilité partielle	7.95	0.72	0.45
Perte d'une unité	3.14	0.48	0.22
Abandon d'un trait. exploitation	3.65	0.42	0.23
Tous modes confondus*	38.2	0.52	0.22

* Pour l'ensemble des données de défaillance relatives au logiciel

Figure V-8 : Taux de défaillance résiduel par mode de défaillance

On remarque que le taux de défaillance relatif à l'indisponibilité générale du sous-système correspond au taux le plus faible. En revanche les défaillances entraînant une indisponibilité partielle ont le taux le plus élevé.

En faisant la somme des taux de défaillances par mode on trouve :

$$\sum_{k=1}^{4} \lambda_{sk} = 15{,}9\ 10^{-7}\ h^{-1}$$

Cette valeur est pratiquement égale à la moitié de celle obtenue à partir de l'application directe du modèle sur l'ensemble des données de défaillance :

$$\lambda_S = 38{,}2\ 10^{-7}\ h^{-1}.$$

Ceci pourrait provenir du petit nombre de données par mode, notamment celles relatives à l'indisponibilité générale. Il est cependant intéressant de constater que l'ordre de grandeur est malgré tout conservé.

V-1-2-4 Application des modèles par composant

Afin de réaliser un compromis entre le nombre de données disponibles par composant et une modélisation détaillée du logiciel, les composants considérés correspondent aux quatre fonctions que doit réaliser le système :

- TELEPHONIE,
- EXPLOITATION (gestion des abonnés et de la maintenance),
- DEFENSE (mécanismes de tolérance aux fautes matérielles),
- EXECUTIF.

Le volume (compté en octets) occupé par ces composants ainsi que la répartition des 136 corrections (l'information composant concerné est donnée par les fiches de correction) est la suivante :

- TELEPHONIE : occupe 33% du volume logiciel et contenait 29% des fautes,
- EXPLOITATION : occupe 28% du volume logiciel et contenait 26% des fautes,
- DEFENSE : occupe 26% du volume logiciel et contenait 30% des fautes,
- EXECUTIF : occupe 13% du volume logiciel et contenait 15% des fautes.

Les résultats de l'application du modèle hyperexponentiel aux différents composants et à l'ensemble des données de correction sont résumés par la figure V-9.

En faisant la somme des taux de défaillances par composant on trouve :

$$\sum_{k=1}^{4} \lambda_{sk} = 65{,}8\ 10^{-7}\ h^{-1}$$

Cette valeur est légèrement supérieure à celle obtenue à partir de l'application directe du modèle sur l'ensemble des données de correction :

$$\lambda_S = 47{,}5\ 10^{-7}\ h^{-1}.$$

Composant	$\lambda_S(10^{-7}\ h^{-1})$	RES	DKS
TELEPHONIE	7.5	0.77	0.33
EXPLOITATION	18.8	0.75	0.28
DEFENSE	27.4	0.58	0.26
EXECUTIF	12.1	-0.06	0.17
LOGICIEL**	47.5	0.84	0.42

** Pour l'ensemble des données de correction relatives au logiciel

Figure V-9 : Taux de défaillance résiduel par composant

Cette différence est due essentiellement au fait que les composants EXPLOITATION et EXECUTIF n'ont pas eu une croissance de fiabilité monotone sur toute la période de trois ans alors que les deux autres composants et le logiciel dans sa totalité ont eu une croissance de fiabilité monotone. En fait le composant le plus pénalisant à ce niveau est le composant EXPLOITATION qui a une courbe intensité de défaillance en forme de cloche due à une décroissance de fiabilité les six premiers mois. En répartissant les données en deux paliers correspondant respectivement à six mois et deux ans et demi chacun, les taux de défaillance résiduels pour les composants EXPLOITATION et EXECUTIF deviennent respectivement : $\lambda_{sEXP} = 7{,}3\ 10^{-7}\ h^{-1}$ et $\lambda_{sEXE} = 8{,}3\ 10^{-7}\ h^{-1}$, les autres restant pratiquement inchangés. L'égalité est ainsi pratiquement réalisée (aux incertitudes près, incertitudes dues à l'application des modèles et à la précision des calculs).

Ces résultats montrent que le composant DEFENSE est le composant le moins "fiable" puisqu'il contribue à plus de la moitié du taux de défaillance global. Les trois autres composants ont à peu près le même taux de défaillance. Ceci implique que les mécanismes de tolérance aux fautes matérielles constituent un point dur pour la fiabilité du système et qu'une attention toute particulière doit leur être accordée lors du développement. Ce résultat doit être modulé par le fait que les défaillances du composant DEFENSE n'entraînent une indisponibilité générale que dans 20% des cas, dans 40% des cas elles entraînent la perte d'une unité matérielle et que l'abandon d'un traitement d'exploitation et l'indisponibilité partielle contribuent à 20% chacune.

Remarque:

Le taux de défaillance résiduel en vie opérationnelle du logiciel évalué à partir des données de correction ($\lambda_S = 4{,}8\ 10^{-6}\ h^{-1}$) est plus élevé que celui évalué à partir des données de défaillance ($\lambda_S = 3{,}8\ 10^{-6}\ h^{-1}$). Ceci est dû au fait que toutes les défaillances ne sont pas

enregistrées alors que toutes les corrections donnent lieu à la rédaction d'une fiche, ce qui donne l'impression que la fréquence des corrections est plus élevée que celle des défaillances. L'écart entre les deux n'est cependant pas très élevé et l'ordre de grandeur est conservé.

V-1-2-5 Conclusions

L'application des modèles de croissance de fiabilité aux données de défaillance et de correction de ce sous-système montre la nécessité d'effectuer les tests de tendance avant de les appliquer. La répartition des données en paliers en fonction des résultats des tests de tendance permet d'obtenir de meilleurs résultats aussi bien en appliquant les modèles sur l'ensemble du logiciel que par composant.

L'application du modèle hyperexponentiel par mode de défaillance a montré que la contribution des défaillances entraînant une indisponibilité générale est faible comparée à celles des autres modes de défaillance.

L'application de ce même modèle par composant a permis de constater qu'il est nécessaire de porter les efforts sur le composant DEFENSE puisque son taux de défaillance résiduel est relativement élevé comparé aux autres composants.

Il serait souhaitable d'effectuer la double décomposition des données par composant et par conséquence afin d'évaluer la contribution de chaque composant à chaque mode de défaillance et surtout de dégager les liens éventuels entre composant et mode de défaillance. Ceci ne peut être possible que dans le cas où on dispose d'informations plus complètes sur les données de défaillance et où il y a correspondance entre défaillance et correction, ce qui n'est pas le cas pour ce logiciel.

V-2- AUTOCOMMUTATEUR TELEPHONIQUE TROPICO-R

Il s'agit d'un autocommutateur téléphonique permettant de raccorder 1500 abonnés [Mel 86, Via 88]. C'est un système distribué dont certains sous-ensembles sont tolérants aux fautes matérielles. Les données de défaillances ont été collectées aussi bien en phases de validation pré-opérationnelle (test de validation et test sur site) qu'en vie opérationnelle. Le logiciel, écrit en Assembleur, compte environ 300 kilo-octets. La durée de la validation était d'environ dix mois, le test sur site a pris environ quatre mois et nous avons pris en compte les données ayant eu lieu les treize premiers mois de la vie opérationnelle (époque où a eu lieu l'étude de fiabilité).

Les données de fiabilité sont enregistrées sur des rapports de défaillance. Il n'y a que les défaillances qui ont pu être simulées qui donnent lieu à la rédaction d'un rapport.
Les redécouvertes ne sont pas enregistrées : si plusieurs rapports issus de différents groupes de test ou de plusieurs sites donnent lieu aux mêmes corrections, un seul rapport de défaillance est enregistré au niveau de la base de données. En fait au niveau de la base de données un rapport de défaillance constitue à la fois le rapport de défaillance et le rapport de correction et contient les informations suivantes :
- la date d'occurrence de la défaillance,
- la configuration du système et les conditions d'occurrence des défaillances,
- le type de défaillance : matérielle, logicielle et les composants affectés,
- les modifications effectuées, le contrôle de ces modifications et les tests de régression correspondants.

Un total de 461 rapports ont été enregistrés, répartis comme suit :
- 297, durant la validation,
- 55, durant le test sur site,
- 109, en vie opérationnelle.

Le nombre de données étant relativement élevé, nous avons préféré travailler dans un premier temps sur des données groupées : l'unité de temps adoptée correspond à 10 jours. Ceci permet de prévoir le nombre de corrections qui devront être effectuées afin de planifier le personnel nécessaire pour le test en phase de validation pré-opérationnelle et pour la maintenance en vie opérationnelle.
La figure V-10 donne le nombre de défaillances par unité de temps.

Le test des données douteuses revêt moins d'importance quand on considère le nombre de défaillances par unité de temps ; seuls les tests de tendance seront appliqués.

Unité de temps	Validation	Test sur Site	vie op.
1	7	4	4
2	8	5	15
3	36	13	21
4	45	20	21
5	60	22	26
6	74	26	29
7	82	27	31
8	98	41	32
9	106	45	32
10	115	48	35
11	120	53	35
12	134	55	35
13	139	-	36
14	142	-	41
15	145	-	46
16	153	-	48
17	167	-	55
18	174	-	61
19	183	-	62
20	196	-	65
21	200	-	67
22	214	-	68
23	223	-	77
24	246	-	88
25	257	-	91
26	277	-	96
27	283	-	102
28	286	-	104
29	292	-	104
30	297	-	104
31	-	-	105
32	-	-	106
33	-	-	107
34	-	-	107
35	-	-	107
36	-	-	108
37	-	-	108
38	-	-	108
39	-	-	109

Figure V-10 : Nombre de défaillances par unité de temps

V-2-1 Tests de tendance

Le test de Laplace appliqué à l'ensemble des données indique une forte croissance de fiabilité globale : $u(80) \simeq -12$. L'application de ce test pas à pas permet de distinguer cinq périodes (Figure V-11) :

- une période allant jusqu'à l'unité de temps 14 où la fiabilité a tendance à croître progressivement,

- une période de décroissance locale malgré une croissance globale de fiabilité allant jusqu'à la 25 ième unité de temps,
- une période de croissance de fiabilité avec des fluctuations locales allant jusqu'à l'unité 53,
- une période de quasi-stabilité (ou légère décroissance) entre la 54 ième et la 70 ième données,
- une période de croissance de fiabilité.

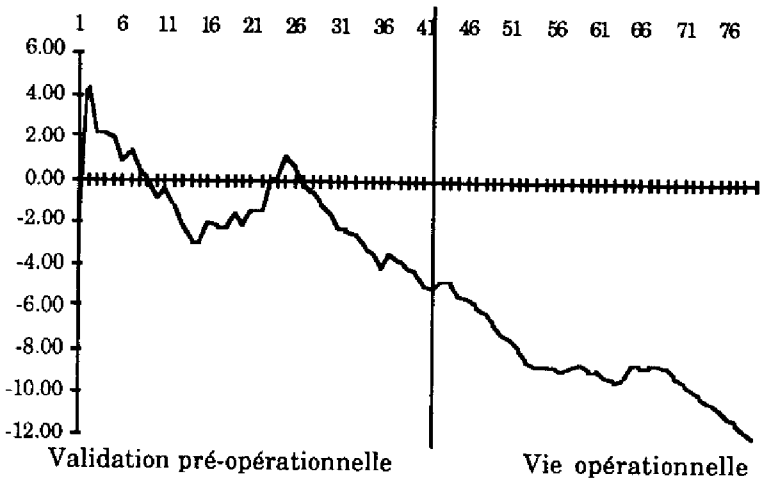


Figure V-11 : Test de Laplace appliqué à toutes les données

La décroissance de fiabilité au tout début de la période d'observation est due à l'activation de 28 fautes durant la troisième unité de temps alors qu'il n'y a eu que 8 fautes activées durant les deux premières unités de temps et 24 dans les deux unités suivantes. Si on ne tient pas compte des deux premières unités de temps et qu'on applique le test de Laplace à nouveau, le coefficient de tendance devient négatif sur toute la période restante. Les résultats sont illustrés à la figure V-12 ; on remarque que le sens d'évolution de la fiabilité donné par cette figure n'a pas changé par rapport à celui indiqué par la figure précédente.

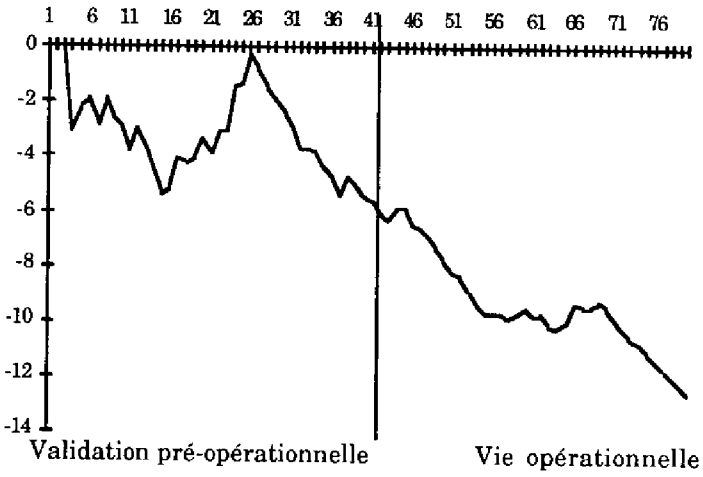


Figure V-12 : Test de Laplace appliqué à partir de la troisième donnée

Le résultat de l'application du test de Laplace par phase est donné à la figure V-13 , il montre que, si on ne considère que les données relatives à la vie opérationnelle, la période comprise entre la 54 et 70 ième unité de temps correspond à une forte décroissance de fiabilité locale.

Il est intéressant de remarquer que le passage du test de validation au test sur site n'a pas entraîné de discontinuité au niveau de l'évolution de la fiabilité ; de même le passage du test sur site à la vie opérationnelle ne s'est pas accompagné d'une forte discontinuité non plus. Les points d'inflexion sont situés à l'intérieur d'une même phase.
La discontinuité durant la validation est essentiellement due au changement de la nature des tests appliqués au logiciel qui correspondent, dans l'ordre respectivement : test fonctionnel, test de qualité, test de performance et test de surcharge.

La discontinuité en vie opérationnelle a été entraînée principalement par une augmentation du nombre de sites qui est passé progressivement de 4 à 15 (figure V-14). Vers l'unité de temps 70 une nouvelle version du logiciel, permettant de raccorder 4096 (au lieu de 1500) abonnés, a été introduite et à partir de cette période aucune version 1500 n'a été installée. Ceci explique la croissance de fiabilité observée à partir de cette période.

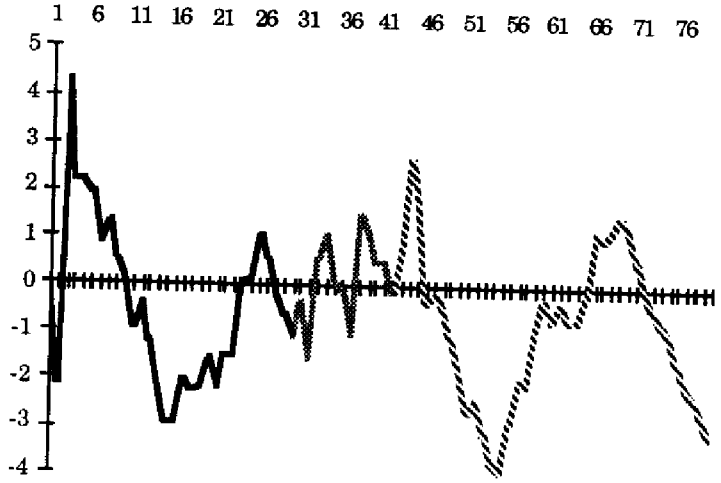


Figure V-13 : Test de Laplace appliqué par phase

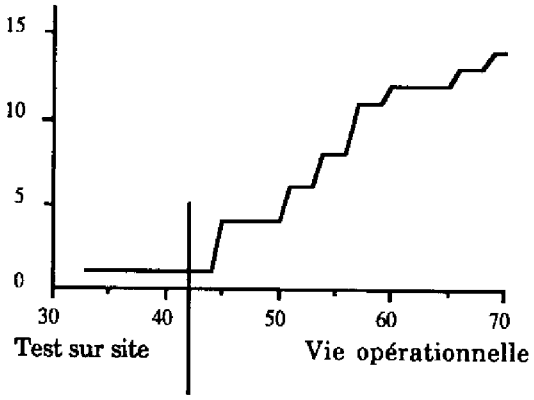


Figure V-14 : Evolution du nombre de sites en vie opérationnelle.

V-2-2 Application des modèles

On dispose de toutes les données relatives aux tests de validation et sur site ainsi que des données relatives à la première année de la vie opérationnelle. On pourrait appliquer les mêmes modèles de croissance de fiabilité que pour le cas précédent, partitionner les données et appliquer à nouveau les modèles par palier ou par fenêtre. Il nous a semblé préférable de suivre une approche légèrement différente afin d'illustrer comment on peut utiliser les modèles de croissance de fiabilité comme aide à la planification du développement et de la maintenance en vie opérationnelle. Nous prendrons d'abord comme variable aléatoire le nombre cumulé de défaillances. Etant donné que les tests de tendance ont montré l'existence de zones de décroissance de fiabilité, nous appliquerons un modèle en forme de S : S-Shaped model (SS). Le modèle hyperexponentiel sera ensuite appliqué aux données, intervalles entre défaillances, de la vie opérationnelle afin d'évaluer le taux de défaillance résiduel du logiciel.

V-2-2-1 Nombre cumulé de défaillances

Le résultat de l'application du modèle SS aux données du test de validation pré-opérationnelle est illustré à la figure V-15 où le nombre cumulé de défaillances entre la 9 ième et la 30 ième unité de temps est estimé à partir des 8 premières données. On voit que ce nombre est sous-estimé à partir de la 15 ième unité de temps, or nous avons déjà constaté au paragraphe précédent que ce point correspond à un point de changement de tendance (point d'inflexion de type 2). Le modèle localise mal ce point et ne tient pas compte de cette croissance subite du nombre de défaillances (décroissance de fiabilité).

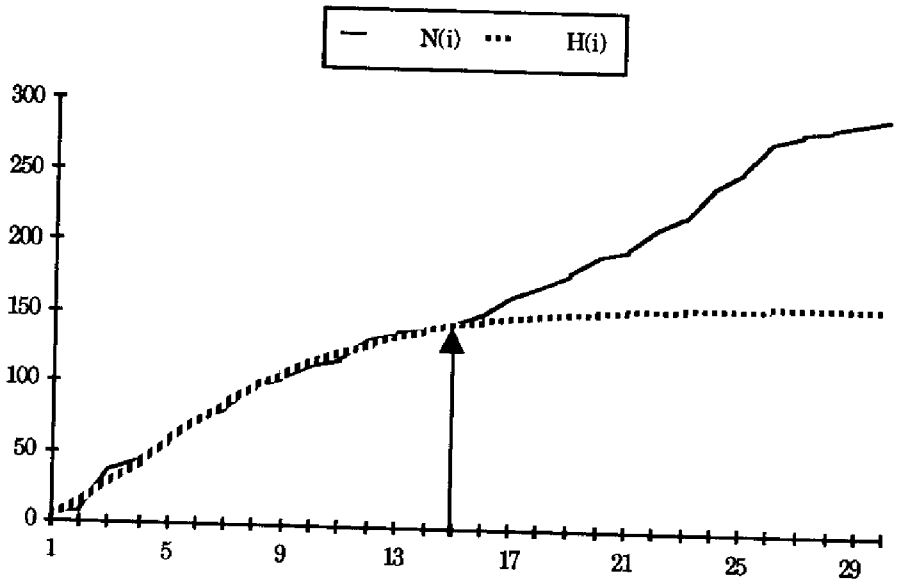


Figure V-15 : TROPICO-R : Nombre cumulé de défaillances en validation

De la même façon si on applique ce modèle aux données de la vie opérationnelle, les estimations du nombre cumulé de défaillances sont très mauvaises à partir de la 55 ième unité de temps. Ces résultas sont donnés à la figure V-16 où le nombre cumulé de défaillances à partir de la 51 ième unité de temps est estimé à l'aide des données 43 à 50. Le point 55 est aussi un point d'inflexion de type 2.

D'après ces deux évaluations rapides nous voyons que les deux points d'inflexion de type 2 constituent un obstacle à l'obtention de bons résultats. Nous avons alors effectué une partition des données selon quatre paliers définis comme suit :

- palier P1 : regroupant les données 1 à 14,
- palier P2 : regroupant les données 15 à 42,
- palier P3 : correspondant aux données 43 à 54,
- palier P4 : correspondant aux données 55 à 81.

Les deux premiers paliers sont relatifs à la phase de test de validation pré-opérationnelle. Les paliers 1 et 3 correspondent à des zones de croissance locale de fiabilité situées après des zones de décroissance globale et contiennent chacun un point d'inflexion de type 1 (Cf. figure V-11). Les paliers 2 et 4 correspondent à une décroissance de fiabilité suivie d'une croissance et semblent se prêter à une évaluation par un modèle en forme de S. Les points d'inflexion de type 2 constituent les frontières P1 $\rightarrow$ P2 et P3 $\rightarrow$ P4.

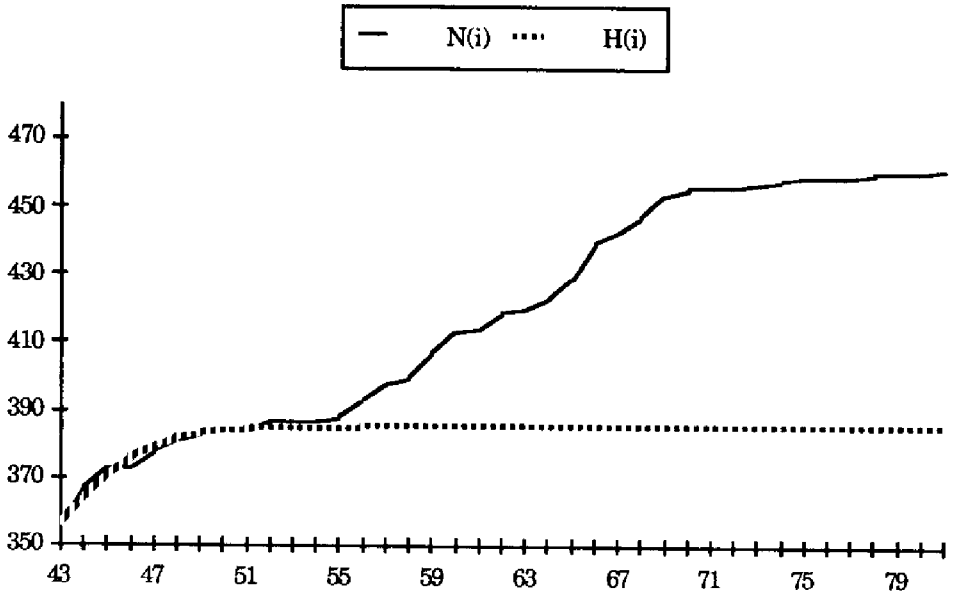
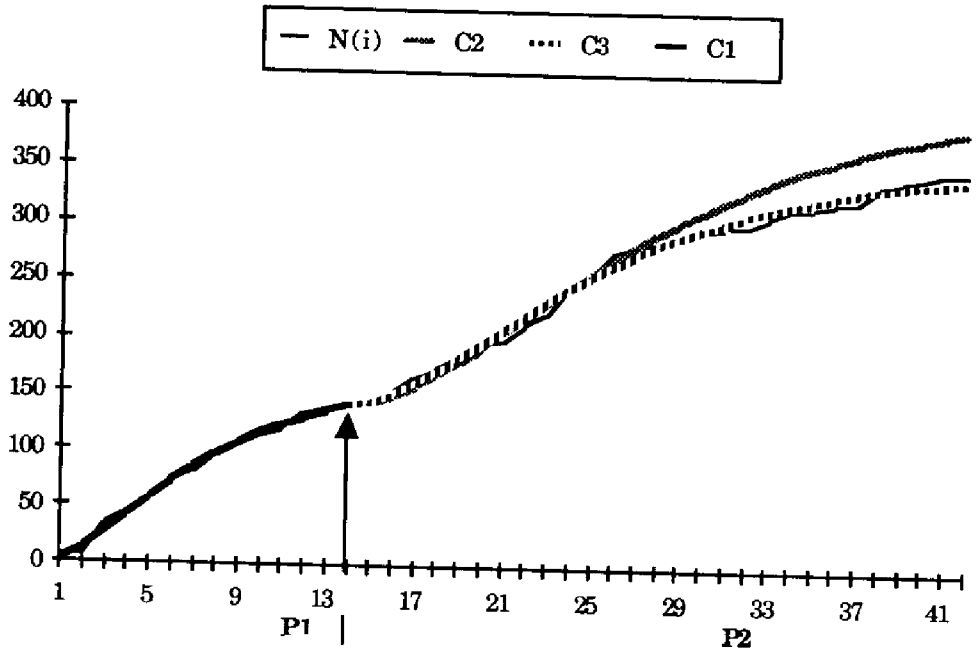


Figure V-16 : Nombre cumulé de défaillances pour la vie opérationnelle

Le modèle SS est alors appliqué sur chacun de ces paliers séparément. Pour P1 et P3 les évaluations précédentes restent valables.
En ce qui concerne P2, on utilise les données 15 à 27 pour calibrer le modèle et estimer le nombre cumulé de défaillances jusqu'à la 42 ième unité de temps. Les estimations commencent à diverger à partir de la 28 ième unité de temps, ce qui prouve que le choix de l'intervalle d'estimation n'est pas optimal. Les évaluations sont représentées sur la courbe C2 de la figure V-17.

Cette courbe montre cependant qu'à la fin du test de validation (unité 30) le logiciel n'aura pas atteint un état de stabilité et que même à la fin du test sur site cet état ne sera pas atteint non plus : H(t) continue à croître. Ceci a été confirmé par la suite par les données relatives au début de la vie opérationnelle. Au vu de ces résultats, l'équipe de développement aurait pu être capable de conclure que si la phase de validation n'est pas accélérée, le logiciel serait livré avant que les programmes de test en cours n'aient atteint leur limite. L'estimation du nombre cumulé de défaillances était entachée d'erreurs mais les aspects "qualitatifs" de la prévision sont tout à fait adéquats.

Pour ce même palier, si le modèle est calibré à l'aide des données 15 à 29, l'évaluation prévisionnelle du nombre cumulé de défaillances est beaucoup plus proche de la réalité ; ceci est illustré par la courbe C3 de la figure V-17.



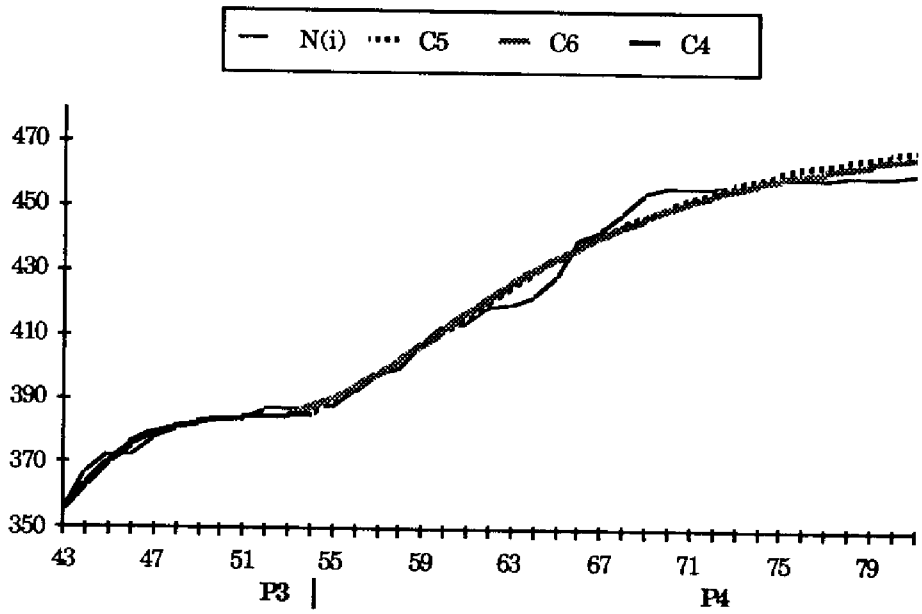
	$E_{9,14}$	$E_{28,42}$	$E_{30,42}$
C1	2.6		
C2		28.4	31.2
C3			5.8

Figure V-17 : Nombre cumulé de défaillances pour les paliers P1 et P2.

A titre indicatif, on donne dans le tableau des figures V-17 et V-18 les valeurs de l'écart moyen entre l'estimation et l'observation : $E_{k,p}$ donné par l'expression IV-22. L'écart entre C2 et C3 est dû uniquement au fait que le point 25 est un point d'inflexion (de type 1), c'est-à-dire un point au delà duquel l'allure de l'évolution de la fiabilité va changer. C2 se base sur les données allant jusqu'à deux observations au delà du point d'inflexion alors que C3 est basée sur les données allant jusqu'à 4 points au delà. Deux points ne sont donc pas suffisants (dans ce cas particulier) pour que le modèle intègre les dernières données et suive la nouvelle tendance.

En ce qui concerne le palier P4, bien que le système soit en vie opérationnelle, on observe une décroissance de fiabilité jusqu'à la 71 ième donnée. On pourrait —bien que le but en vie opérationnelle soit différent— continuer à prévoir l'évolution du nombre cumulé de défaillances afin de prévoir d'une part le moment où le logiciel aura atteint un régime stable et d'estimer la charge de maintenance qu'il faut continuer à assurer d'autre part. L'application du modèle SS basée sur les données 54 à 73 permet de tracer la courbe C5 de la figure V-18. La croissance de fiabilité semble linéaire à partir de la 75 ième unité de temps, les estimations prévoient donc un comportement stabilisé à partir de la 75 ième unité de temps soit

deux mois avant la fin de la période d'observation en vie opérationnelle. En fait d'après les données enregistrées seules deux défaillances ont été observées pendant les deux derniers mois. La courbe C6 est obtenue à partir des données 54 à 75 et est pratiquement confondue avec C5.



	$E_{51,55}$	$E_{74,81}$	$E_{76,81}$
C4	1.8		
C5		4.3	5.3
C6			3.5

Figure V-18 : Nombre cumulé de défaillances pour les paliers P3 et P4.

L'application du modèle SS jusqu'à l'unité de temps 90 (trois mois au delà de la période considérée) indique que pour l'ensemble des sites installés, le nombre estimé de défaillances est de deux défaillances le mois suivant et une défaillance par mois pour les deux mois suivants, soit quatre défaillance au cours du trimestre suivant.

Remarque

Une autre partition en trois paliers (indépendante du cycle de vie) aurait pu être effectuée : P2 et P3 pourraient être regroupés en un seul palier puisque u(k) continue à décroître de façon monotone même après le passage à la vie opérationnelle (Cf. figure V-11). L'estimation de P3 à partir des données de P2 est tout à fait convenable.

V-2-2-2 Taux de défaillance en vie opérationnelle

Pour la vie opérationnelle si on veut évaluer les temps moyens entre défaillances ou le taux résiduel de défaillance, il est plus judicieux d'appliquer le modèle hyperexponentiel (HE).

Le modèle SS a été appliqué à l'ensemble des données collectées sur tous les sites afin d'estimer le nombre de corrections à effectuer sur le logiciel installé sur tous les sites. Ces résultats sont surtout intéressants pour le concepteur du logiciel à des fins de gestion de la maintenance.

L'utilisateur est plutôt intéressé par la fiabilité d'un site afin d'évaluer soit le nombre de fois où le système va être réinitialisé suite à une défaillance du logiciel, soit le nombre d'appels à maintenance. Les intervalles entre défaillances à considérer sont donc ceux relatifs à un site moyen. Les intervalles entre défaillances (ramenés à un site) observés durant les derniers mois de la vie opérationnelle sont donnés par la figure V-19 (à lire de gauche à droite).

14	14	0*	0	14	14	42	0	0	14	0	0
42	42	14	0	90	15	0	15	0	120	15	90
0	0	0	90	30	420	180	90	555	480		

* k zéros successifs indiquent que (k+1) défaillances ont eu lieu le même jour.

Figure V-19 : Intervalles entre défaillances (en jours) pour les derniers mois de vie opérationnelle considérés.

Test de Laplace:

Les résultats de l'application du test de Laplace à ces données sont tracés sur la figure V-20. Ce test indique une période où la fiabilité est pratiquement stable (u(k) oscille entre -2 et +2) suivie d'une période de croissance de fiabilité effective. La croissance de fiabilité à la fin de la période est essentiellement due aux 5 dernières observations dont l'intervalle moyen est de l'ordre de 345 jours pour un site moyen alors qu'il est de 24 jours pour les 5 observations précédentes. Ces intervalles sont respectivement 23 jours et 1,6 jour pour l'ensemble des 15 sites installés.

Application du modèle HE :

Les intervalles entre défaillances observés (t_i) et estimés par HE ($MTTF_i$) sont données à la figure V-21. On peut remarquer que les 5 dernières estimations sont très

mauvaises et que le résidu ($R_{k,p}$ donné par l'expression V-16) passe de 0,65 à 0,10 si on arrête l'estimation avant ces cinq valeurs.

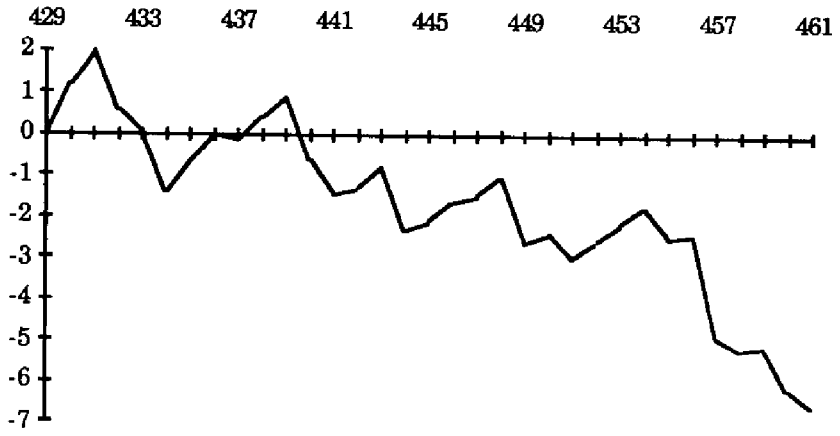


Figure V-20 : Facteur de Laplace pour la période considérée.

i	t_i	$MTTF_i$
441	42	23.35
442	14	38.52
443	0	28.77
444	90	18.77
445	15	36.05
446	0	32.23
447	15	27.23
448	0	25.56
449	120	22.41
450	15	32.87
451	90	31.16
452	0	36.22
453	0	33.38
454	0	30.91
455	90	28.77
456	30	32.71
457	420	32.54
458	180	54.78
459	90	61.64
460	555	62.81
461	480	88.03

$R_{441,461} = 0.65$ $R_{441,456} = 0.10$

$\lambda_s = 8.3\ 10^{-3}$/jour

Figure V-21 : Résultats d'application du modèle hyperexponentiel.

Le taux de défaillance résiduel estimé par le modèle est de 3,4 10^{-4}/h (8,3 10^{-3}/jour) ce qui correspond à une défaillance en moyenne par site tous les 4 mois. Cet intervalle est beaucoup plus faible que celui qui a été effectivement observé.

Le taux de défaillance résiduel estimé sans tenir compte des cinq dernières données est de 1,25 10^{-3}/h (3 10^{-2}/jour). Ce taux paraît relativement élevé. Une façon d'utiliser ces résultats est la suivante : si l'utilisateur constate que ce taux est très élevé et ne correspond pas aux spécifications, il pourrait demander au concepteur d'améliorer la fiabilité du logiciel. Ce dernier doit alors développer des tests supplémentaires à appliquer de nouveau au logiciel. Ces tests pourront alors être appliqués aussi bien sur site, en parallèle avec l'exploitation normale du système, que hors ligne sur une copie du logiciel. Une façon d'améliorer le taux de défaillance du logiciel serait d'utiliser des techniques de tolérances aux fautes simples telles que celles décrites dans [Gra 86] ou [Mou 85] ou de rendre certaines composants du logiciel tolérants aux fautes [Ran 75, Arl 88].

On remarque à nouveau que le modèle HE n'arrive pas à suivre une croissance de fiabilité très brutale ; il sur-estime le taux de défaillance résiduel (et sous-estime le MTTF) en vie opérationnelle : c'est un modèle qui est assez pessimiste dans ce genre de situation.

Afin de surmonter cette difficulté, les données sont réparties en deux paliers comme suit :

P5 : données comprises entre les indices 427 et 447 pour lesquelles il n'y a pas de croissance de fiabilité évidente,

P6 : données à partir de l'indice 448 pour lesquelles la croissance de fiabilité est nette.

Application du modèle HE au palier P6 :

Les résultats de l'application de HE pas à pas aux données de P6, visualisés à la figure V-22, sont meilleurs que les précédents. Les prévisions des 5 derniers intervalles entre défaillances sont meilleures et le taux de défaillance résiduel correspond tout à fait à celui observé à la fin de la période considérée et même plus tard. Ce taux est de 1,3 10^{-4}/h (2,98 10^{-3}/jour), ce qui correspond à un intervalle entre défaillances de 336 jours pour un site, soit une défaillance en moyenne tous les 22,4 jours pour les 15 sites ou 4 défaillances par trimestre pour les 15 sites.

Remarque :

Si on considère le taux de défaillance du système (matériel et logiciel), la contribution du logiciel semble prépondérante puisque le taux de défaillance dû au matériel est de l'ordre

de 4 10^{-6}/h alors que celui du logiciel est de l'ordre de 10^{-4}/h. En fait ces deux valeurs ne sont pas tout à fait comparables puisque :

- le matériel est en partie tolérant aux fautes et le taux de 4 10^{-6}/h correspond aux défaillances qui entraînent une indisponibilité du système,
- le taux de défaillance du logiciel est évalué toutes conséquences confondues, or la majorité des défaillances n'entraînent pas l'indisponibilité du système ou la perte de communications.

Il faudrait effectuer l'étude de fiabilité du logiciel en tenant compte des différents modes de défaillances. Malheureusement les données collectées ne sont pas suffisamment détaillées pour fournir ce genre d'information.

i	t_i	$MTTF_i$
454	0	37.50
455	90	32.14
456	30	39.37
457	420	38.33
458	180	323.89
459	90	279.01
460	555	216.17
461	480	298.43

$R_{454,456} = 0.31$

$\lambda_s = 2.98\ 10^{-3}$/jour

Figure V-22 : Résultats d'application du modèle hyperexponentiel à P6.

V-3- CONCLUSIONS

L'étude de la fiabilité de ces autocommutateurs permet d'illustrer les différents aspects de la méthode d'évaluation de la fiabilité du logiciel proposée dans ce mémoire.
Les tests de tendance appliqués aux données aident à partitionner les données collectées sur le logiciel en paliers pour lesquels la tendance est homogène et correspond soit à une croissance de fiabilité soit à une décroissance de fiabilité suivie d'une croissance de fiabilité.
Pour chacun des systèmes nous avons appliqué les modèles de croissance de fiabilité en faisant abstraction de toute étude de tendance et en utilisant toutes les données collectées. L'application des modèles de croissance de fiabilité en tenant compte des résultats des tests de tendance (par palier ou par fenêtre) améliore considérablement les prévisions.

L'application du modèle hyperexponentiel par composant et par mode de défaillance à un sous-ensemble de l'autocommutateur E-10 a permis de constater que :

- le taux de défaillance le plus élevé en vie opérationnelle correspond au composant logiciel chargé de gérer la tolérance aux fautes du matériel,
- le taux de défaillance le plus faible en vie opérationnelle correspond au mode de défaillance le plus critique : indisponibilité générale.

L'application conjointe de deux modèles de nature différente aux données de défaillance de l'autocommutateur TROPICO-R (un modèle en forme de S et le modèle hyperexponentiel respectivement) a permis d'une part d'estimer le nombre de corrections qu'il faut continuer à assurer en vie opérationnelle et d'évaluer le taux de défaillance résiduel du logiciel d'autre part.

Appliquée à ces deux logiciels différents, la méthode a prouvé qu'elle possède un caractère général et qu'elle peut ainsi être appliquée à n'importe quel type de logiciel ... du moins à des logiciels possédant des caractéristiques analogues : une partie temps réel de traitement (ou de contrôle ou de commande) et une partie type gestion qui traite de l'exploitation du système. Cette méthode a l'avantage d'entraîner une réflexion approfondie concernant le système étudié, elle procure ainsi une meilleure connaissance du système et permet d'avoir confiance dans les résultats des évaluations de fiabilité.

Les deux systèmes considérés correspondent tous deux au même type d'application : commutation téléphonique et concernent par conséquent des applications non critiques. Après défaillance du logiciel, le système est réinitialisé de façon automatique, et les durées de réinitialisation sont très courtes. Il s'en suit que les durées d'arrêt sont très courtes et que l'indisponibilité due aux défaillances du logiciel est négligeable par rapport à celle entraînée par les défaillances du matériel.

CONCLUSION GENERALE

Les résultats exposés dans ce mémoire peuvent être classés en deux catégories :

- modélisation de la sûreté de fonctionnement du logiciel,
- méthode globale d'évaluation de la fiabilité d'un logiciel et application à l'évaluation de la sûreté de fonctionnement de deux autocommutateurs téléphoniques.

Concernant la modélisation du logiciel, l'approche que nous avons proposée et suivie est une approche descendante et progressive, elle peut se résumer par les points suivants :

- définition de politiques de correction se rapprochant le plus possible de celles qui correspondent à celles suivies par les utilisateurs et élaboration du modèle de sûreté de fonctionnement correspondant : modèle de connaissance,
- établissement, à l'aide des processus de renouvellement généralisés, de l'expression de l'intensité de défaillance (ou de la disponibilité) relative à ces modèles,
- approximation du modèle de connaissance par un modèle d'action permettant de traduire la même évolution des mesures de sûreté de fonctionnement.

Les modèles de connaissance sont aptes à suivre n'importe quelle évolution de "fiabilité" : croissance, décroissance et croissance suivie de décroissance ou inversement. Par contre les modèles d'action qui ont été développés dans la littérature ne peuvent suivre qu'une croissance de fiabilité ou une décroissance de fiabilité. Afin de surmonter cette difficulté nous répartissons les données de défaillance collectées sur le logiciel en sous-ensembles (appelés paliers) présentant l'une des caractéristiques précédentes. Cette partition des données est effectuée à l'aide des tests de tendance utilisés dans le cadre de la méthode d'évaluation de la fiabilité du logiciel que nous proposons.

Concernant la méthode d'évaluation proposée, outre la possibilité de détecter l'existence d'éventuelles données douteuses et de délimiter les paliers de croissance et de décroissance de fiabilité, elle permet d'aboutir à des résultats plus significatifs puisque les modèles de croissance de fiabilité sont appliqués à des données plus "homogènes".

Dans cette méthode, le suivi de l'évolution du nombre cumulé de défaillances (qui est en soi un test de tendance) constitue un critère de choix pour :

- changer de jeu de test afin d'activer des erreurs latentes non activables par le jeu de test en cours,
- arrêter le test du logiciel si tous les jeux de test sont épuisés ... ou développer de nouveaux jeux de test.

Deux mesures complémentaires peuvent être évaluées afin de répondre aux différents objectifs du concepteur et de l'utilisateur :

- comme l'un des buts est d'estimer la fréquence des interventions afin d'évaluer l'effort de test et de maintenance qu'il faut assurer d'une part et le nombre de relances que doit subir le système d'autre part, il est donc intéressant d'évaluer le nombre de défaillances pour une période future,
- dans le but d'avoir une estimation du taux de défaillance global du système matériel et logiciel, l'utilisateur est aussi intéressé par le taux de défaillance (ou le MTTF) .

L'étude de deux exemples d'applications particulières de la méthode montre qu'il est possible, avec la méthode proposée, de répondre à ces objectifs. Les résultats obtenus, bien que partiels, servent d'ores et déjà à nos partenaires ALCATEL-CIT et TELEBRAS-CPqD (tant la méthode que les résultats) :

- pour ALCATEL-CIT, les résultats ont permis d'effectuer une évaluation globale de la sûreté de fonctionnement du système en vie opérationnelle à l'aide de données réellement observées sur le système.
- pour TELEBRAS-CPqD, un nouvel autocommutateur est en cours de développement et la même méthode lui sera appliquée assez tôt afin de pouvoir avoir un retour rapide vers les équipes de réalisation du logiciel et influencer le déroulement du développement. L'évaluation de la sûreté de fonctionnement de l'ensemble du système à l'aide des résultats de notre étude (et des évaluations analogues concernant le matériel) est en cours de réalisation à TELEBRAS-CPqD.

Le travail effectué nous paraît riche d'enseignements sur deux plans au moins :

- d'un strict point de vue scientifique, la rigueur de la démarche aussi bien pour l'étude de l'influence des politiques de correction que pour la méthode d'évaluation,
- d'un point de vue de l'utilisation de la méthode d'évaluation et des résultats par le secteur socio-économique.

L'ensemble des résultats exposés dans ce mémoire est bien entendu susceptible de nombreuses extensions. En particulier trois points semblent nécessiter des développements :

- tests de tendance avec des niveaux de confiance,
- critères de validation de modèles de croissance de fiabilité plus adaptés à la croissance de fiabilité,
- évaluation de la fiabilité d'un logiciel en fonction de celles de ses composants en fiabilité évolutive et en tenant compte des dépendances éventuelles entre les différents composants.

Le cas de la disponibilité du logiciel risque de poser des problèmes à long terme. Certes les temps d'arrêt dus au logiciel peuvent être courts comparés aux durées d'arrêt entraînées par le matériel, dans le cas où la relance est possible. En revanche le nombre d'arrêts suite à défaillance du logiciel risque d'être plus élevé que le nombre d'arrêts dus au matériel. Ainsi, l'indisponibilité d'un système due au logiciel peut devenir du même ordre de grandeur que celle due au matériel. Il faut donc être vigilant sur ce point et une collecte de données semble être nécessaire afin de confirmer (ou infirmer) aussi bien ces propos que le modèle de disponibilité établi dans ce mémoire.

Les travaux effectués dans ce mémoire sont concernés par la sûreté de fonctionnement d'un logiciel qui "existe" et sur lequel on a collecté des données : *évaluation a posteriori*. Il serait intéressant de développer des méthodes permettant d'évaluer la sûreté de fonctionnement d'un logiciel avant même qu'il ne soit codé : *évaluation a priori*. Une méthode d'évaluation a priori peut reposer sur :

- l'observation d'un ensemble de logiciels d'horizons divers,
- l'application de la méthode proposée dans ce mémoire pour l'évaluation des mesures de sûreté de fonctionnement de ces logiciels,
- la mémorisation des résultats et l'établissement de liens et de corrélations entre ces différents logiciels.

On pourrait ainsi essayer de dégager des similitudes entre le comportement des logiciels en fonction du langage utilisé, de la taille du logiciel, du type d'application, des méthodes de codage et de test utilisées au cours du développement... afin d'aboutir à des recommandations (et peut être même à des normes). Une telle méthode nécessite l'observation d'un grand nombre de logiciels, et encore une fois la collecte de données est au cœur du problème.

ANNEXE 1

PROPRIETES DES PROCESSUS

DE POISSON NON HOMOGENES

Distribution du nombre cumulé de défaillances

Soit $P_n(t)$ la probabilité que le nombre d'événements sur l'intervalle [0,t] soit égal à n étant donné qu'à l'instant t_0 le nombre d'événements est égal à n_0 :

$$P_n(t) = Pr\{N(t) = n \mid N(t_0) = n_0\}$$

Le but est d'exprimer $P_n(t)$ en fonction de l'intensité de défaillance h(t).

$$\begin{aligned} P_n(t+dt) = {} & Pr\{N(t) = n \mid N(t_0) = n_0 \text{ et aucun événement entre t et t+dt}\} \\ & + Pr\{N(t) = (n-1) \mid N(t_0) = n_0 \text{ et 1 événement entre t et t+dt}\} \\ & + \sum_{k=2}^{n-n_0} Pr\{N(t) = (n-k) \mid N(t_0) = n_0 \text{ et k événements entre t et t+dt}\} \end{aligned}$$

Soit :

$$P_n(t+dt) = P_n(t)\,P_0(dt) + P_{n-1}(t)\,P_1(dt) + \sum_{k=2}^{n-n_0} P_{n-k}(t)\,P_k(dt) \qquad (A1.1)$$

avec : $P_j(dt) = Pr\{N(t+dt) - N(t) = j\}$

En utilisant les propriétés d'un processus de Poisson, on a :

$$P_0(dt) = 1 - h(t)\,dt + o(dt)$$
$$P_1(dt) = h(t)\,dt + o(dt)$$
$$P_k(dt) = o(dt) \qquad \text{pour } k > 1$$

Il vient :

$$P_n(t+dt) = P_n(t)\,[1 - h(t)\,dt] + P_{n-1}(t)\,h(t)\,dt + o(dt) \qquad (A1.2)$$

d'où :

$$P_n(t+dt) - P_n(t) = [- h(t) P_n(t) + h(t) P_{n-1}(t)]\, dt + o(dt)$$

Si $P'_n(t)$ est la dérivée par rapport au temps de $P_n(t)$ on a :

$$P'_n(t) = - h(t) P_n(t) + h(t) P_{n-1}(t) \qquad \text{pour } n > n_0$$

Ce système d'équations peut être résolu de façon récursive, $P_{n0+1}(t)$ est solution de :

$$P'_{n0+1}(t) = - h(t) P_{n0+1}(t) + h(t) P_{n0}(t)$$

avec $P_{n0}(t)$ obtenu à partir de :

$$P'_{n0}(t) = - P_{n0}(t)\, h(dt),$$

ce qui donne :

$$P_{n0}(t) = \exp-[H(t)-H(t_0)]$$

d'où :

$$P_{n0+1}(t) = [H(t)-H(t_0)] \exp-[H(t)-H(t_0)]$$

On suppose que $P_{n-1}(t)$ est de la forme :

$$P_{n-1}(t) = \frac{[H(t)-H(t_0)]^{n-1-n_0}}{(n-1-n_0)!} \exp-[H(t)-H(t_0)]$$

$P_n(t)$ est alors solution de :

$$P'_n(t) = - h(t) P_n(t) + h(t) \frac{[H(t)-H(t_0)]^{n-1-n_0}}{(n-1-n_0)!} \exp-[H(t)-H(t_0)]$$

Après intégration, on obtient :

$$\Pr\{N(t) = n \mid N(t_0) = n_0\} = \frac{[H(t)-H(t_0)]^{n-n_0}}{(n-n_0)!} \exp-[H(t)-H(t_0)] \qquad \text{(A1.3)}$$

Fonction de survie (ou de fiabilité)

La fonction de fiabilité est définie comme étant la probabilité de bon fonctionnement continu du logiciel entre $S_{i-1}=s$ (instant d'occurrence du i-1 ième événement) et t.

Soient :

- X_i, la variable aléatoire intervalle jusqu'au i ième événement,
- S_i, l'instant d'occurrence du i ième événement,
- $t'=t-s$.

On a :

$$R(t' \mid s) = Pr\{X_i > t' \mid S_{i-1}=s\}$$

$$\begin{aligned} R(t \mid s) &= Pr\{S_i > t \mid S_{i-1}=s\} \\ &= 1 - Pr\{S_i \le t \mid N(s)=i-1\} \\ &= 1 - Pr\{N(t) \ge i \mid N(s)=i-1\} \\ &= 1 - \sum_{j=i}^{\infty} Pr\{N(t) - N(s)=j-i\} \end{aligned}$$

Utilisant (A1.3), on obtient :

$$R(t \mid s) = \exp-[H(t)-H(s)]$$

ou : $R(t' \mid s) = \exp-[H(s+t')-H(s)]$

L'expression de la fiabilité permet d'obtenir directement, à l'aide du tableau de la figure I-4, les autres attributs de la sûreté de fonctionnement.

Fonction densité de probabilité

Par définition :

$$f(t' \mid s) = \frac{d\,F(t' \mid s)}{dt} = \frac{d\,[1 - R(t' \mid s)]}{dt} = h(s+t')\,R(t' \mid s)$$

Soit, en fonction de t :

$$f(t \mid s) = h(t)\,R(t \mid s)$$

Taux de défaillance

$$\lambda(t \mid s) = \frac{d\,F(t' \mid s)}{dt\,R(t' \mid s)} = \frac{f(t' \mid s)}{R(t' \mid s)} = h(s+t')$$

Soit, en fonction de t :

$$\lambda(t \mid s) = h(t)$$

ANNEXE 2

ETABLISSEMENT DE L'EXPRESSION DU FACTEUR DE LAPLACE

L'expression du facteur de Laplace u(k) quand la variable aléatoire considérée est le nombre cumulé de défaillances est établie comme indiqué dans [Cox 78, p. 54].

Considérons k intervalles de temps de longueur l et soient :

N_i la variable aléatoire représentant le nombre de défaillances dans [(i-1)l,il], i=1,...,k,

n_i la réalisation de N_i, c'est-à-dire le nombre de défaillances dans [(i-1)l,il], i=1,...,k,

Supposons que le processus de défaillance est un processus de Poisson Non Homogène (NHPP) d'intensité :

$$h(t) = e^{a+bt} \qquad \text{(A2.1)}$$

Pour b=0 le processus de Poisson devient homogène et l'intensité de défaillance est indépendante du temps.

La fonction de vraisemblance relative aux k intervalles de temps est :

$$L(k) = \prod_{i=1}^{k} P\{N_i = n_i\} = \prod_{i=1}^{k} \frac{\left[\int_{(i-1)l}^{il} h(u)\,du\right]^{n_i}}{n_i!} \exp\left(-\int_{(i-1)l}^{il} h(u)du\right) \qquad \text{(A2.2)}$$

En remplaçant h(u) par (A2.1), cette fonction devient :

$$L(k) = \frac{(e^{bl}-1)^N\, e^{aN+bl\sum_{i=1}^{k}(i-1)n_i}}{b^N \prod_{i=1}^{k} n_i!} \exp\left[\frac{e^a(1-e^{bkl})}{b}\right], \text{ avec } N=\sum_{i=1}^{k} n_i \qquad \text{(A2.3)}$$

La fonction densité de probabilité conditionnelle (sachant que N défaillances ont eu lieu sur les k intervalles de temps) est obtenue en divisant l'expression (A2.3) par la probabilité :

$$P(k) = P\{\sum_{i=1}^{k} n_i = N\} = \frac{\left[\int_{0}^{kl} h(u)\,du\right]^N}{N!} \exp\left(-\int_{0}^{kl} h(u)du\right)$$

Le résultat ne dépend que de la variable b et est donné par (A2.4) :

$$\frac{L(k)}{P(k)} = \frac{N!\,(e^{bl}-1)^N\, e^{bl\sum_{i=1}^{k}(i-1)n_i}}{(e^{bkl}-1)^N \prod_{i=1}^{k} n_i!} \qquad \text{(A2.4)}$$

dont le logarithme $\mathcal{L}$ (b) est :

$$\mathcal{L}(b) = \log N! - \log \prod_{i=1}^{k} n_i! + bl \sum_{i=1}^{k}(i-1)n_i + N\left[\log(e^{bl}-1) - \log(e^{bkl}-1)\right]$$

La dérivée de cette fonction est :

$$\mathcal{L}'(b) = \frac{d\mathcal{L}(b)}{db} = \begin{cases} h\sum_{i=1}^{k}(i-1)n_i + N\left[\frac{le^{bl}}{e^{bl}-1} - \frac{kle^{bkl}}{e^{bkl}-1}\right] & , b\neq 0 \\ l\sum_{i=1}^{k}(i-1)n_i - \frac{Nl(k-1)}{2} & , b=0 \end{cases} \qquad \text{(A2.5)}$$

et la fonction information $\mathcal{I}$ (b), est donnée par :

$$\mathcal{I}(b) = E\{-\mathcal{L}''(b)\} = \begin{cases} Nl^2\left[\frac{e^{bl}}{(e^{bl}-1)^2} - \frac{k^2e^{bkl}}{(e^{bkl}-1)^2}\right] & , b\neq 0 \\ \frac{Nl^2(k^2-1)}{12} & , b=0 \end{cases} \qquad \text{(A2.6)}$$

Pour b=0 (intensité de défaillance constante), la statistique :

$$u(k) = \frac{\mathcal{L}'(0)}{\sqrt{\mathcal{I}(0)}} = \frac{\frac{\sum_{i=1}^{k}(i-1)n_i}{N} - \frac{(k-1)}{2}}{\sqrt{\frac{(k^2-1)}{12N}}}$$

a une distribution normale de moyenne zéro et de variance unité.

Les valeurs positives de u(k) correspondent à une intensité de défaillances croissante dans le temps et sont obtenues pour b>0. A l'inverse, les valeurs négatives de u(k) sont obtenues pour b<0 et correspondent à une intensité de défaillance décroissante (fiabilité croissante).

ANNEXE 3

INTENSITE DE DEFAILLANCE D'UN LOGICIEL EN FONCTION DES INTENSITES DE DEFAILLANCE PAR MODE DE DEFAILLANCE

Afin d'établir l'expression de l'intensité de défaillance en fonction de celles par mode de défaillance, nous nous servons à nouveau des modèles de connaissance utilisés au premier chapitre. Pour un mode de défaillance donné, le processus de défaillance peut être modélisé par un processus de Poisson par morceaux. A tout instant le processus de défaillance du logiciel résulte de la superposition de processus de Poisson homogènes (puisque entre deux défaillances successives chaque processus est à taux constant). Entre deux défaillances, le processus résultant est donc un processus de Poisson homogène dont le taux est égal à la somme des taux selon les différents modes de défaillance. La figure A3-a illustre ce phénomène pour deux modes de défaillances : A et B ; cette figure correspond au cas où les deux processus sont à taux de défaillance strictement décroissant alors que dans la figure A3-b le processus de défaillance correspondant au cas où le mode A passe par une croissance locale tout en ayant un taux globalement décroissant. Cette croissance locale se retrouve au niveau du processus résultant et peut être rapidement atténuée (ou accentuée) par la décroissance (resp. croissance) due aux autres modes de défaillance.

L'expression de l'intensité de défaillance est établie en considérant le processus de renouvellement généralisé correspondant à $\lambda(t)$.Le taux de défaillance est égal à la somme des taux de défaillance selon les différents modes de défaillance et, comme nous l'avons vu au second chapitre, l'allure de l'intensité de défaillance suit celle des taux de défaillance : il en résulte que l'allure de l'évolution de l'intensité de défaillance est directement liée à l'évolution de celles relatives aux différents modes de défaillance.

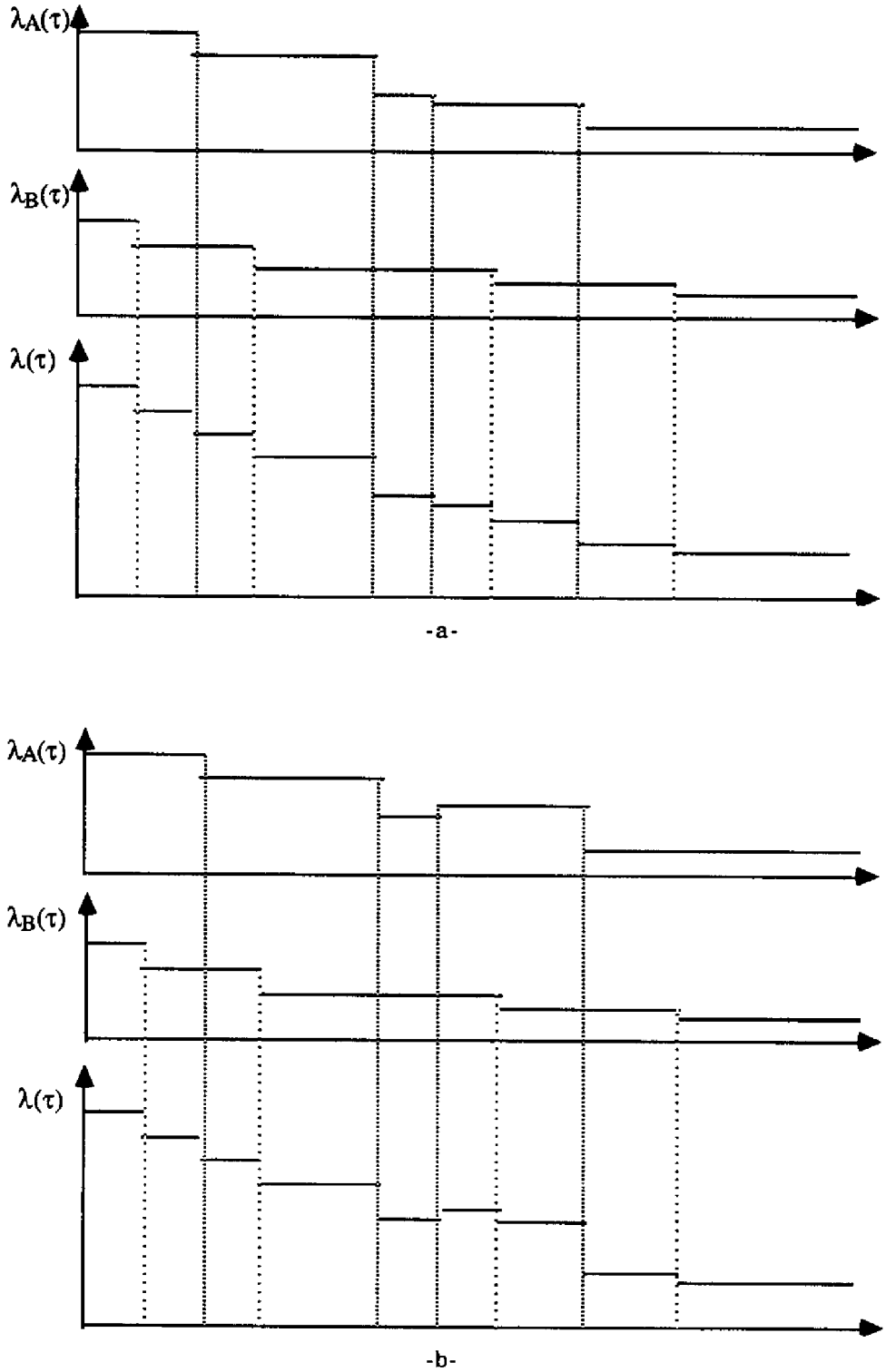


Figure A3: Taux de défaillance d'un logiciel en fonction des taux de défaillance par mode de défaillance.

REFERENCES

Abd 86 A.A.Abdel-Ghaly, P.Y.Chan, B.Littlewood : "Evaluation of Competing Software Reliability Predictions", *IEEE Trans. on Soft. Eng.*, vol. SE-12, n°9, Septembre 1986, pp. 950-967.

Ada 80 E.N.Adams : "Minimizing cost impact of software defects", IBM Research Division, Report RC 8228 (35669), Avril 1980.

Afn 85 Proposition de norme AFNOR, *Traitement de l'information, vocabulaire de la qualité du logiciel*, référence Z61-102, Juillet 1985.

Aïv 70 S.Aïvazian : *Etude Statistique des Dépendances*, Editions MIR Moscou 1970.

Arl 88 J.Arlat, K.Kanoun, J.C.Laprie : "Dependability Evaluation of Software Fault Tolerance", Acte du 18th Int. Symp. on *Fault-Tolerant Comp. (FTCS-18)*, Tokyo, Japon, 27-30 Juin 1988, pp. 142-147.

Asc 78 H.Ascher, H.Feingold : "Application of Laplace's Test to Repairable System Reliability", Acte du *1er Colloque international sur la fiabilité et la maintenabilité*, Paris, 19-23 Juin 1978, pp. 219-225.

Asc 84 H.Ascher, H.Feingold : *Repairable systems reliability*, Lecture notes in statistics, vol. 7, New York and Basel : Marcel Dekker, inc, 1984.

Bak 88 T.C.Baker : "Effect of Field Service on Software maintenance", *IEEE Trans. on Soft. Eng.* vol. SE-14, n° 2, Février 1988, pp. 254-258.

Bar 75 R.E.Barlow, F.Proschan : *Statistical Theory of Reliability and Life Testing*, Edition Holt, Rnehart and Winston, Inc, 1975.

Bas 88 a M.R.Bastos Martini, J.Moreira de Souza, H.M.F.Tavares : "Central téléphonique TROPICO-R : contrôle de la qualité du logiciel", Acte du *6ième Colloque international sur la Fiabilité et la Maintenabilité*, Strasbourg, 3-7 Octobre 1988, pp. 213-218.

Bas 88 b M.R.Bastos Martini, K.Kanoun, J.Moreira de Souza : "Software Reliability Evaluation of the TROPICO-R Switching System". Rapport de recherche LAAS n° 88-336, Novembre 1988.

Basi 84 V.R.Basili, D.M.Weiss : "Methodology for Collecting Valid Software Engineering Data", *IEEE Trans. on Soft. Eng.* vol. SE-10, n° 6, Novembre 1984, pp. 728-738.

Basi 88 V.R.Basili : "A study on Fault Prediction and Reliability Assessment in the Nasa/SEL Environment", Acte de *Annual National Joint Conference and Tutorials on Software Quality and Reliability*, Virginie, USA, 1-3 Mars 1988, pp. 359-366.

Bra 68 J.V.Bradly : *Distribution free statistical tests*, Edition Printice Hall Incorporation, 1968.

Boe 76 B.W.Boehm : "Software Engineering", *IEEE Transactions on Computers*, vol. C-25, n°12, Décembre 1976, pp. 1226-1241.

Boe 81 B.W.Boehm : *Software Engineering Economics*, Prentice Hall Inc, Englewood Cliffs, N.J., 1981.

Cha 86 P.Y.Chan : "Software reliability prediction", thèse de P.H.D., Department of mathematics, The City University, Londres, 1986.

Cor 75 M.Corazza : *Techniques mathématiques de la fiabilité des systèmes,* édition Cepadues 1975.

Cos 78 A.Costes, C.Landrault, J.C.Laprie : "Reliability and Availability models for maintained systems featuring hardware failures and software faults", *IEEE Trans. on Computers*, vol. C-27, Juin 1978, pp. 548-560.

Cox 66 D.R.Cox, P.A.W.Lewis : *The Statistical Analysis of Series of Events*, Londres, Chapman & Hall, 1966.

Cox 68 D.R.Cox, H.D.Miller : *The theory of stochastic processes*, Londres, Methuen 1968.

Cox 70 D.R.Cox : *Renewal Theory*, Methuen' Monographs on Applied Probabibity and Statistics 1970.

Cro 77 L.H.Crow : "Confidence interval procedures for reliability growth analysis", Tech. report 1977, US Army Material Syst. Anal. Activity Aberdeen, MD, 1977.

Cur 86 P.A.Currit, M.Dyer, H.D.Mills : " Certifying the reliability of software", *IEEE Tran. on Software Eng.*, vol. SE-12, n° 1, Janvier 1986, pp. 3-11.

Dal 86 C.J.Dale : "Software Reliability Models", *Pergamon Infotech State of the art Report*, Pergamon Infotech, UK, Avril 1986, pp. 31-44 et 244-247.

Daw 84 A.P.Dawid : "Statistical theory : The prequential approach", *J.Roy. Statist. Soc.* A, vol. 147, 1984, pp. 274-292.

Des 82 Y.Deswarte et al. : "ARMURE : architecture multiprocesseur redondante du système informatique de SARGOS", Acte du *3ième Colloque international de Fiabilité et de Maintenabilité"*, Toulouse, 18-21 Octobre 1982, pp. 103-111.

Dua 64 J.T.Duane : "Learning curve a pproach to reliability monitoring", *IEEE Trans. on Aerospace*, vol. 2, 1964, pp. 563-566.

Fav 85 J.Favrot, C.Lamy, F.Michel : "Mesure de la fiabilité d'un logiciel en qualification pour le contrôle des combustibles nucléaires irradiés", *Technique et Science Informatiques*, vol. 4, n°2, 1985, pp. 209-223.

Fle 64 R.Fletcher, M.C.Reeves : "Function minimization by gonjugate gradients", *Comput. J.*, 7, 1964, pp. 149-154.

Fin 79 J.M.Finkelstein : "Starting & Limiting Values for Reliability Growth", *IEEE Trans. on Reliability*, vol. R-28 n° 2, Juin 1979, pp. 111-113.

Fon 85 V.Font : "Une approche de la fiabilité des logiciels : modèles classiques et modèle linéaire généralisé", thèse de troisième cycle, Université Paul Sabatier, Toulouse, Septembre 1985.

Fuk 86 K.Fukishima, Y.Kishida : "Estimation of the bug curve using experimental regression analysis of office automation equipement software", Acte du *5ième Colloque international de Fiabilité et de Maintenabilité*", Biarritz, 6-10 Octobre 1986, pp. 87-91.

Gau 88 O.Gaudoin : "Les tests de tendance de fiabilité des systèmes réparables, application à la fiabilité du logiciel", Rapport Technique IMAG-TIM3, n° 41, Juillet 1988.

Gau 89a O.Gaudoin : "Deux ou trois choses que je sais du test de Laplace", Rapport Technique IMAG-TIM3, n° 48, Avril 1989.

Gau 89b O.Gaudoin, J.L. Soler : "Analyse statistique d'un modèle de fiabilité des logiciels", Rapport de Recherche IMAG-TIM3, n° 781-M, Juin 1989.

Gla 81 R.L.Glass : "Persistent Software Errors", *IEEE Trans. on Soft. Eng.*, vol. SE-7, n° 2, Mars 1981, pp. 162-168.

Gné 72 B.Gnédenko, Y.K.Béliaev, A.D.Soloviev : *Méthodes mathématiques en théorie de la fiabilité*, Mir 1972.

Goe 79 A.L.Goel, K.Okumoto : "Time-dependent Error-detection Rate Model for Software and other Performance Measures", *IEEE Trans. on Reliability*, vol. R-28, n°3, Août 1979, pp. 206-211.

Goe 83 A.L.Goel : "A guidebook for Software reliability assessment", Data and Analysis Centre for Software, *Rome Air Development Centre (RADC)*, Rome, New York, Tech. Report RADC-TR-83-176, 1983.

Goe 85 A.L.Goel : "Software Reliability Models : Assumptions, Limitations, and Applicability" *IEEE Trans. on Software Eng., Special issue on software Reliability*, vol. SE-11, n° 12, Décembre 1985, pp. 1411-1423.

Gra 86 J.Gray : "Why do computers stop and what can be done about it ?" 5th Symposium on *Reliability in Distributed Software and database Systems*, Janvier 1986, pp. 3-12.

Hol 74 M.Hollander, F.Prochan : " A test for superadditivity for the mean value function of a Non Homogeneous Poisson Process", *Stoch. Proc. and their a pplication* , vol. 2, 1974, pp. 195-209.

Hop 80 A.L.Hopkins, Jr : " Fault-Tolerant System Design : Broad Brush and Fine Print", *IEEE Computer* , Mars 1980, pp. 39-45.

Iye 82 R.K.Iyer, S.E.Butner, E.J.Mac Cluskey : "A Statistical Failure Load Relationship: Results of a Multicomputer Study", *IEEE Trans. on Comp.*, vol. C-31, n° 5, 1982, pp. 697-706.

Jel 72 Z.Jelinski, P.B.Moranda : "Software Reliability Research", Acte de *Statistical Methods for the Evaluation of Computer System Performance*, Academic Press, 1972, pp. 465-484.

Kaâ 89 M.Kâaniche, K.Kanoun, S.Metge : " Proposition d'une procédure de collecte de données pour l'évaluation de la sûreté de fonctionnement des systèmes informatiques", Rapport de recherche LAAS n° 89-329, Septembre 1989.

Kan 85 K.Kanoun, J.C.Laprie : "Modeling Software Reliability and Availability from Development-validation up to Operation", Rapport de recherche LAAS n° 85-042, Mars 1985, révision Août 1985.

Kan 86 K.Kanoun : "Validation de Modèles Stochastiques, application aux Modèles de Croissance de Fiabilité du Logiciel", Rapport de Recherche LAAS n° 86-002, Janvier 1986.

Kan 87 a K.Kanoun, J.C.Laprie, T.Sabourin : "Définition d'une approche globale pour l'évaluation de la fiabilité du logiciel", Rapport de recherche LAAS, n° 87-096. Mars 1987, Révision Décembre 1987.

Kan 87 b K.Kanoun, T.Sabourin : "Software Dependability of a Telephone Switching System", Acte du 17th Int. Symp. *Fault-Tolerant Comp. (FTCS-17)*, Pittsburgh, Pennsylvania, 6-8 Juillet 1987, pp. 236-241.

Kan 87 c K.Kanoun, T.Sabourin : "Analyse des défaillances et évaluation de la fiabilité du logiciel d'un autocommutateur téléphonique", *Technique et Science Informatiques* (TSI), Vol. 6, n° 4, 1987, pp. 287-303.
Rapport de recherche LAAS n° 86-362, Décembre 1986.

Kan 88 a K.Kanoun, J.C.Laprie, T.Sabourin : "A method for Software Reliability Growth Analysis and Evaluation", Acte du : *Le Génie Logiciel & ses Applications*, Toulouse, 5-9 Décembre 1988, pp. 859-878.
Rapport de recherche LAAS n° 88-170, Juin 1988.

Kan 88 b K.Kanoun : "Analyse de la croissance de fiabilité du logiciel", Acte du *6ième Colloque international sur la Fiabilité et la Maintenabilité*, Strasbourg, 3-7 Octobre 1988, pp. 651-656.
Rapport de recherche LAAS n° 88-171, Juin 1988.

Kan 88 c K.Kanoun : "Modèles de croissance de fiabilité du logiciel : une approche incorporant les changements de spécifications", Rapport de contrat Aérospatiale-LAAS n° 26152-5289.
Rapport de recherche LAAS n° 88-172, Juin 1988.

Kan 88 d K.Kanoun, M.R.Bastos Martini, J.Moreira de Souza, : "A method for Software Reliability Analysis and Prediction, Application to the TROPICO-R Switching System", rapport de Recherche LAAS, n° 88-337, Décembre 1988, révision Avril 1989.

Kei 83 P.A.Keiller, B.Littlewood, D.R.Miller, A.Sofer : "Comparison of Software Reliability Predictions", Acte du *13th Int. Symp. Fault-Tolerant Comp.* (FTCS 13), Milan, Italie, Juin 28-30 1983, pp. 128- 134.

Lal 88 J.H.Lala, L.S.Alger : "Hardware and Software Fault Tolerance : A Unified Architectural Approach", Acte du 18th Int. Symp. on *Fault-Tolerant Comp. (FTCS-18)*, Tokyo, Japon, 27-30 Juin 1988, pp. 240-245.

Lap 75 J.C.Laprie : "*Prévision de la sûreté de fonctionnement et architecture de structures numériques temps réel réparables*", thèse de doctorat ès-Sciences, Université Paul Sabatier Toulouse, Juin 1975.

Lap 83 J.C.Laprie : "Evaluation de la sûreté de fonctionnement des logiciels en opération", *Technique et Science Informatiques*, vol.2, n°4, Décembre 1983, pp. 233-247.

Lap 84 a J.C.Laprie : "Dependability Modeling and Evaluation of Hardware-and-Software Systems", Acte du *2nd GI-NTG-GMR Conf. on Fault Tolerant Computing,* Bonn, Allemagne Fédérale, Septembre 1984, pp. 202-215.

Lap 84 b J.C.Laprie : "Dependability Evaluation of Software Systems in Operation", *IEEE Trans. on Soft. Eng.* Vol. SE-10, n° 6, Novembre 1984, pp. 701-714.

Lap 88 J.C.Laprie : "Vers une théorie de la fiabilité du X-iel", *Technique et Science Informatiques*, Vol. 7, n° 3, 1988, pp. 315-330.

Lap 89 J.C.Laprie : "Dependability : a unifying concept for reliable computing and fault tolerance", *Dependability of Resilient Computers,* Editor : T.Anderson, BSP Professional Books, 1989, pp. 1-28.

Law 82 J.F.Lawless : *Statistical models and methods for lifetime data,* Wiley series in probability and mathematical statistics, 1982.

Leh 80 M.M.Lehman : "Programs, Life Cycles, and Laws of Software Evol.ution", *Proceedings of the IEEE*, vol. 68, n°9, Septembre 1980.

Leh 85 M.M.Lehman, L.A.Belady : *Program Evol.ution, Processes of Software Changes,* Academic Press 1985.

Lit 73 B.Littlewood, J.L.Verral : "A Bayesian Reliability Growth Model for Computer Software" *J. Royal Stat. Soc.*, C(App. stat.), 22, 1973, pp. 332-336.

Lit 79 B.Littlewood : "Software Reliability model for modular program structure", *IEEE Trans. on Reliability,* vol. R-28, n° 3, Août 1979, pp. 241-246.

Lit 80 B.Littlewood : "Theories of software Reliability : How good are they and how can they be improved?", *IEEE Trans. on Soft. Eng.* vol. SE-6, n° 5, Septembre 1980, pp. 489-500.

Lit 81 B.Littlewood : "Stochastic Reliability-Growth : A Model for Fault-Removal in Computer-Programs and Hardware-Designs", *IEEE Trans. on Reliability,* vol.-R-30, n° 4, pp. 313-320, 1981.

McC 89 C.McCollin, A.Bendell, D.W.Wightman : "Effects of Explanatory factors on Software Reliability", Acte de *Reliability'89*, 1989, à paraître.

Mel 86 E.B.Mello et al. : "Software Architecture of a TROPICO-R Telephone Exchange", *6 th Inter. Conf. On Software Eng. for Telecom. Switching System*, Londres, Avril 1986, pp. 40-45.

Mel 87 P.Mellor : "Software Reliability modelling : the state of the art", *Information and Software Technology*, vol. 29, n° 2, Mars 1987, pp. 81-98.

Mil 86 D.R.Miller : "Exponential order statistic models for software reliability growth", *IEEE Trans. on Software Eng.*, vol. SE-12, n° 1, 1986, pp. 12-24.

Moa 81 R.Moawad : "Fiabilité du logiciel : Modélisation et évaluation de modèles", thèse de Docteur Ingénieur, ENSAE, Toulouse, Décembre 1981.

Mor 75 P.Moranda : "Predictions of software reliability during debugging", Acte du *Ann. Reliability and Maintainability Symposium*, Washington, D.C., Janvier 1975, pp. 327-332.

Mou 85 S.Mourad, D.Andrews : "Reliability of the IBM MVS/XA Operating System", Acte du 15th Int. Symp. on *Fault-Tolerant Comp. (FTCS-15)*, Ann Arbor, Michigan, Juin 1985, pp. 93-98.

Mus 75 J.D.Musa : "A theory of Software Reliability and its Application", *IEEE Trans. on Soft. Eng.*, vol. SE-1, Septembre 1975, pp. 312-327.

Mus 79 J.D.Musa : "Software reliability data", Data and Analysis Centre for Software, *Rome Air Development Centre (RADC)*, Rome, New York, Tech. Report, 1979.

Mus 84 J.D.Musa, K.Okumoto : "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement", *7th Int. Conf. on Soft. Eng.*, Orlando, Florida, Mars 1984, pp. 230-238.

Mus 87 J.D.Musa, A.Iannino, K.Okumoto : *Software Reliability: Measurement, Prediction, Application*, McGraw-Hill International Editions, Computer Science Series, 1987.

Nag 82 P.M.Nagel, J.A.Skrivan : "Software Reliability : Repetitive Run Experimentation and Modeling", BCS 98124, Boeing Computer Service Company, Seattle, Washington, Février 1982.

Ohb 82 M.Ohba, S.Yamada, K.Tekada, S.Osaki : " S-Shaped software reliability growth curve : how good is it?", Acte du *COMPSAC 82*, Chicago, Illinois, 8-12 Novembre 1982, pp. 38-44.

Ohb 84 M.Ohba, S.Yamada : "Modèles de croissance de fiabilité du logiciel à forme en S", Acte du *4ième Colloque international de Fiabilité et de Maintenabilité*, Perros Guirec, France, 1984, pp. 430-436.

Par 80 F.N.Parr : "An alternative to the Rayleigh Curve Model for Software Development Effort", *IEEE Trans. on Soft. Eng.* vol. SE-6, n° 3, Mai 1980, pp. 291-296.

Per 89 U.D.Perera : "Reliability Demonstration and Field Analysis of a Communications Controller", Acte de *Reliability Data Collection and Use in Risk and Availability*, Sienne, Italie, 15-17 Mars 1989, pp. 871-886.

Pow 64 M.J.D.Powell : "An efficient method for finding the minimum of a function of several variables without calculating derivatives", *Comput. Journal*, vol. 7, n° 2, 1964, 155-162.

Ram 82 C.V.Ramamoorthy, F.B.Bastani : "Software Reliability - Status and Perspectives" *IEEE Trans. on Soft.Eng.*, vol. SE-8, n° 4, Juillet 1982, pp. 354-371.

Ran 75 B. Randell, "System structure for software fault tolerance", *IEEE Trans. on Software Engineering*, vol. SE-1, n° 2, Juin 1975, pp. 220-232.

Rem 82 L.Rémus : "Methodology for software development of a digital integrated protection system", TC 7 Safety and Security meeting, Janvier 1982, Bruxelles.

Rob 83 P.J.Robinson, R.K.Atkins : "Building Reliable Software for Spacelab", Acte de *SAFECOMP'83* Workshop,Cambridge, 20-22 Septembre 1983, pp. 153-158.

Ros 82 D.J.Rossetti, R.K.Iyer : "Software-related Failures on the IBM 3081 : A Relationship with System Utilization", CRC Tech. rep. n° 82.8, juin 1982.

Sab 86 T.Sabourin, K.Kanoun : "Analyse des défaillances et modélisation du comportement du logiciel d'un autocommutateur téléphonique", Acte du *5ième Colloque international de Fiabilité et de Maintenabilité*, Biarritz, 6-10 Octobre 1986, pp. 92-97.

Sab 87 T.Sabourin : "Evaluation de la fiabilité du logiciel d'un autocommutateur téléphonique", thèse de doctorat , INPT, Octobre 1987.

Sap 78 G.Saporta : *Théorie et méthodes de la statistique*, Technip, Paris 1978.

Sch 79 R.E.Schafer et al. : "Validation of software reliability models", *Rome Air Development Centre (RADC)*, Rome, New York, Tech. Report, Juin 1979.

Sho 73 M.Shooman : "Operating testing and software reliability during program development", *IEEE Symp. Comput. Software Rel.*, New York, 30 Avril - 2 Mai, 1973, pp. 51-57.

Sol 88a J.L.Soler : "Remarques sur la fiabilité des systèmes présentant des fautes de conception. Application à la fiabilité des logiciels", Rapport Technique IMAG-TIM3, n° 36, Mai 1988.

Sol 88b J.L.Soler : "Modélisation des processus à risque, de défaillance et de correction de systèmes présentant des fautes de conception. Application à la fiabilité des logiciels", Acte du *6ième Colloque international sur la Fiabilité et la Maintenabilité*, Strasbourg, 3-7 Octobre 1988, pp. 647-650.

Swa 76 E.B.Swanson : "The Dimension of Maintenance", IEEE Computer Society, *2nd International Conference on Software Engineering*, Los Alamitos, 1976, pp. 492-497.

Tri 75 A.K.Trivedi, M.L.Shooman : "A many-state Markov model for the estimation and prediction of computer software performance parameters", Conf. Int. *Reliable Software*, Los Angeles, Avril 1975, pp. 208-220.

Tro 86 R.Troy, Y.Romain : "A Statistical Methodology for the Study of the Software Failure Process and Its A pplication to the ARGOS Centre", *IEEE Trans. on Soft. Eng.*, vol. SE-12, n° 9, Septembre 1986, pp. 968-978.

Via 88 B.Vianna : "R&D at TELEBRAS-CPqD : The TROPICO System", *Int. Conf. on Communications (ICC 88)*, Philadelphie, Juin 1988, pp. 632-636.

Wen 78 J.H.Wensley : "SIFT : Design and Analysis of a fault-Tolerant Computer for Air Craft Control", *Proceedings of the IEEE*, vol. 66, n° 12, Octobre 1978, pp. 1240-1254.

Yam 83 S.Yamada, S.Osaki : "Reliability Growth Modeling for Software Error Detection", *IEEE Trans. on Reliability*, vol. R-32, n° 5, 1983, pp. 475-478.

TABLE DES MATIERES

Thèse de Doctorat d'état de Madame Karama KANOUN
"Croissance de la sûreté de fonctionnement des logiciels, caractérisation-modélisation-évaluation"

RÉSUMÉ
Ce mémoire présente des travaux et résultats concernant la modélisation et l'évaluation de la sûreté de fonctionnement de logiciels, aussi bien sur le plan théorique que sur le plan pratique. Le comportement du logiciel est modélisé en fonction de différentes situations : correction immédiate ou différée par lot, changements de spécifications... Pour chacune des situations un modèle de connaissance est établi. Ces modèles sont ensuite approchés par des modèles d'action qui sont moins détaillés et de ce fait moins complexes.
Une méthode d'évaluation de la sûreté de fonctionnement des logiciels est proposée dans la suite du mémoire et est appliquée à deux logiciels différents. Cette méthode est basée sur une collecte de données de défaillances observées sur le logiciel concerné, le prétraitement de ces données et l'évaluation des mesures de sûreté de fonctionnement du logiciel. Les buts du prétraitement des données sont d'une part de s'assurer que les données collectées constituent un ensemble homogène et d'autre part de déterminer les périodes de décroissance et de croissance de sûreté de fonctionnement afin d'appliquer les modèles sur ces périodes séparément. Les prédictions de fiabilité sont ainsi améliorées de façon significative. Les exemples traités montrent tout l'intérêt de cette méthode.

Mots clés : Croissance de fiabilité, modélisation de la croissance de fiabilité, disponibilité, tests de tendance, évaluation de la sûreté de fonctionnement de logiciels.

"Software Dependability Growth characterization, modeling and evaluation"

ABSTRACT
This thesis presents the results concerning software dependability modeling and evaluation; both theoretical and practical aspects are considered. The software behavior is modeled in different situations: immediate correction of faults, delayed corrections, specification changes... A knowledge model is established for each situation. These models are then approached by action models which are less detailed and thus less complex.
A software dependability evaluation method is proposed in the second part and applied to two different systems. This method is based on the collection and processing of failure data. The data set is first pre-processed so as to be sure that it is homogeneous and to identify periods of time during which reliability is increasing or decreasing. Reliability growth models are then applied to these periods separately. Reliability predictions are thus significantly improved. The relevance of the proposed method is shown through its application to the software of two switching systems.

Key words : Reliability growth, reliability growth modeling, availability, trend tests, software dependability evaluation.