Fiabilité du logiciel : de la collecte des données à l'évaluation probabiliste

Karama Kanoun, Mohamed Kaâniche, Jean-Claude Laprie

Fiabilité du logiciel : de la collecte des données à l'évaluation probabiliste

Karama Kanoun, Mohamed Kaâniche et Jean-Claude Laprie
LAAS - CNRS
7, Avenue du Colonel Roche
31077 Toulouse Cedex (France)
e-mail : kanoun@laas.fr

RESUME. Cet article présente une méthode d'analyse et d'évaluation de la fiabilité du logiciel basée sur le traitement des données de défaillance recueillies sur le logiciel en développement ou en exploitation. Cette méthode est progressive : elle débute par un filtrage des données collectées, se poursuit par des partitionnements des données retenues en fonction de certains critères (sévérité des défaillances, localisation des fautes, etc.), afin de permettre des analyses descriptives, des analyses de tendance et des évaluations probabilistes. L'accent est mis sur le traitement des données en temps réel pour permettre un retour d'expérience rapide. L'intérêt de la méthode est montré à travers son application à des données réelles. Sa mise en œuvre est facilitée par un outil : SoRel dont les caractéristiques sont présentées.

ABSTRACT. This article presents a method for software reliability analysis and evaluation based on processing failure data collected on the software during development or in operation. This method is progressive: filtering of the collected data, partitioning of the retained data into subsets according to some criteria (failure severity, fault location, etc.), to perform descriptive analyses, trend analyses and probabilistic evaluations. Emphasis is put on processing of the data in real time to enable early field feedback. The benefits of this method are shown through its application to real data. To make it easier to process failure data, we developed a tool, SoRel, which is presented in the last section.

MOTS-CLES : Fiabilité du logiciel, croissance de fiabilité, traitement de données.

KEY WORDS: Software reliability, reliability growth, failure data processing.

1. Introduction

L'objectif d'un fournisseur de logiciel est de développer un logiciel fiable, dans les meilleurs délais et facilement maintenable. Afin d'atteindre cet objectif, il doit être capable de a) contrôler efficacement les différentes activités du développement, b) gérer le passage d'une phase à la suivante afin de maîtriser les délais de livraison, c) prévoir le moment où un certain niveau de fiabilité serait atteint (ou, du moins, vérifier si le logiciel satisfait les contraintes de sûreté de fonctionnement

éventuellement fixées par le client), et d) s'assurer que le logiciel est facilement maintenable afin de réduire les efforts de maintenance. L'*utilisateur* est intéressé par un logiciel fiable, livré à temps et à un coût raisonnable. La méthode d'analyse et d'évaluation de la fiabilité du logiciel que nous avons développée aide à atteindre au mieux ces objectifs ; elle permet de mieux satisfaire les besoins du fournisseur et de l'utilisateur en adoptant alternativement le point de vue de l'un et de l'autre.

Bien souvent (si ce n'est pas dans la majorité des expériences publiées, ou simplement rapportées) l'évaluation de la fiabilité du logiciel s'est limitée à l'application d'un ou plusieurs modèles de croissance de fiabilité aux données collectées. Et bien souvent, les résultats étaient décourageants car ils étaient de façon évidente non réalistes. Ceci est dû essentiellement à l'absence de méthode rigoureuse de traitement des données collectées. Il s'avère donc nécessaire de suivre une méthode globale d'analyse des données avant l'évaluation de la fiabilité afin de s'assurer à chaque étape de l'étude de a) la validité des données recueillies, b) la cohérence de celles qui sont retenues et c) l'applicabilité des modèles de croissance de fiabilité.

La méthode présentée dans cet article est basée sur le traitement de données de défaillance collectées sur le logiciel durant le développement ou l'exploitation. Elle est constituée de plusieurs étapes allant du filtrage des données collectées à la quantification. Nous mettons l'accent sur le traitement en temps réel dans un environnement contrôlé. L'analyse de la tendance constitue une étape fondamentale et originale : les indicateurs de tendance aident à contrôler la progression du processus de développement en temps réel et permettent ainsi un retour d'expérience rapide. Cette méthode résulte de notre expérience d'une dizaine d'années relative au traitement des données de fiabilité du logiciel ; elle a ainsi évolué tout au long de cette période et s'est concrétisée par le développement d'un outil logiciel, SoRel, facilitant le traitement des données.

L'article est divisé en sept parties. La deuxième partie donne une vue d'ensemble de la méthode. Les troisième, quatrième et cinquième parties s'intéressent respectivement aux statistiques descriptives, à l'analyse de la tendance et à l'application des modèles de croissance de fiabilité. La sixième partie montre l'intérêt de la méthode à travers le traitement des données d'un système réel. Enfin, la dernière partie présente l'outil SoRel.

2. Présentation de la méthode

L'étude de fiabilité est basée sur l'analyse et le traitement des données de défaillance collectées sur le logiciel. La collecte de données doit être clairement définie depuis le début en précisant explicitement ses objectifs, son organisation, le type de données à collecter, le rôle des intervenants, etc. Il est important qu'elle soit effectuée correctement car la pertinence des résultats dépend de la qualité de ces données. Nous nous intéressons principalement au traitement des données[1]. Dans cette partie, nous analysons les objectifs d'une étude de fiabilité, et nous donnons

[1] De nombreuses publications traitent de la définition de la collecte de données et des problèmes liés à cette collecte, on peut consulter par exemple [BAS 84, COM 89]

une vue globale de la méthode de traitement. Les activités des différentes étapes sont plus détaillées dans les parties suivantes.

2.1. *Objectifs d'une étude de fiabilité*

Les objectifs d'une étude de fiabilité sont directement liés au point de vue considéré (celui du fournisseur ou de l'utilisateur) et à la phase du cycle de vie concernée. Ces objectifs sont généralement exprimés en terme de mesures.

Quand *le logiciel est en développement*, ces mesures concernent :

• l'évolution de la tendance (croissance / décroissance de fiabilité) en réponse à l'effort de test afin de contrôler ces activités,

• l'estimation prévisionnelle du nombre de défaillances afin de planifier l'effort nécessaire (en termes de durée de test et d'effectif de l'équipe de test).

Quand *le logiciel est en opération*, deux types de mesures sont intéressants selon le point de vue considéré :

• du point de vue de l'utilisateur : le temps jusqu'à défaillance ou le taux de défaillance afin d'évaluer la fiabilité du système prenant en compte les défaillances du matériel et celles du logiciel,

• du point de vue du fournisseur : estimation du nombre de défaillances en provenance de tous les utilisateurs (ou du nombre de corrections à effectuer) afin d'estimer l'effort de maintenance nécessaire pour répondre aux sollicitations.

Ces objectifs déterminent le type de données à collecter et les traitements à effectuer sur ces données. Deux catégories de données peuvent être recueillies : a) des données caractérisant certains attributs du produit analysé, de son processus de production et de son environnement d'utilisation (taille du logiciel, langage, fonctions, version utilisée, méthodes de vérification et de validation appliquées, profil de test, etc.), et b) des données relatives aux défaillances et modifications (date d'occurrence, nature des défaillances, conséquences, type de fautes, localisation des fautes, etc.). Généralement, ces informations sont indiquées sur des rapports de défaillance et de correction.

2.2. *Vue d'ensemble*

La figure 1 résume les différentes opérations qui peuvent être effectuées sur les données collectées et les types de résultats attendus. Les données recueillis doivent faire l'objet d'une analyse préliminaire approfondie qui vise étudier leur validité éventuellement écarter toutes les données qui ne sont pas pertinentes pour l'étude de fiabilité menée ; un filtrage est souvent nécessaire afin de ne garder que les données qui sont relatives au logiciel. En fonction des objectifs de l'étude, les analyses et évaluations de la fiabilité peuvent être conduites sur l'ensemble des données retenu ou sur des sous ensembles obtenus par partitionnement permettant des analyses plus fines.

Trois types de traitement peuvent être effectués sur ces données : des analyses descriptives, l'analyse de la tendance et l'application de modèles de croissance de fiabilité.

• Les analyses descriptives sont basées sur des statistiques concernant par exemple la densité de fautes, le nombre de fautes par version, des analyses combinées portant sur la localisation des fautes et les conséquences des défaillances, etc. Ces analyses sont généralement de nature qualitative et ne conduisent pas à des évaluations quantitatives de la fiabilité du logiciel. Cependant elles sont très utiles car elles fournissent des indicateurs permettant de mieux comprendre les mécanismes de création et d'élimination des fautes et d'analyser de façon comparative la qualité du produit en cours de développement et de son processus de production par rapport à des produits antérieurs.
• L'analyse de la tendance permet de suivre l'évolution dans le temps des mesures caractéristiques de la fiabilité du logiciel, par exemple le temps jusqu'à occurrence d'une défaillance ou le nombre de défaillances. Elle constitue un moyen puissant pour contrôler la progression du développement et apprécier l'efficacité des procédures de test d'une part, et d'autre part elle permet de guider l'application des modèles de croissance de fiabilité et d'améliorer les prévisions par l'identification des sous ensembles des données pour lesquels les hypothèses des modèles choisis sont vérifiées.
• L'application d'un ou de plusieurs modèles de croissance de fiabilité permet d'obtenir des prévisions des mesures probabilistes de la fiabilité du logiciel à partir des données observées.

Dans la suite du papier, nous présentons plus en détail les différentes étapes de la méthode en mettant l'accent plus particulièrement sur l'analyse de la tendance et l'application des modèles de croissance de fiabilité.

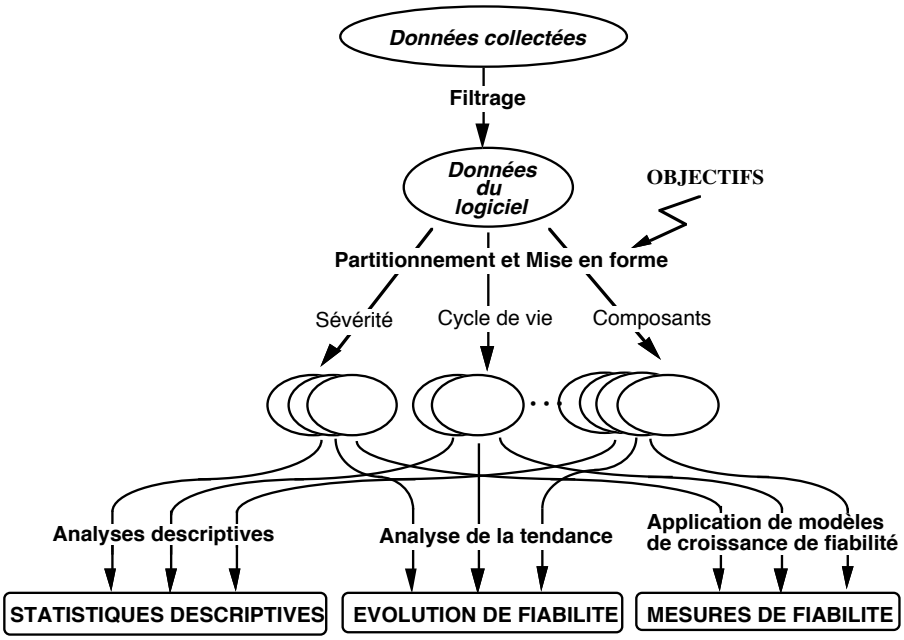


Figure 1. *Étapes successives d'une étude de fiabilité*

Notons enfin que la particularité de notre approche par rapport aux travaux publiés dans le domaine réside dans sa progression et dans l'intégration de différentes techniques complémentaires pour le suivi et l'évaluation de la fiabilité du logiciel. Les travaux publiés s'intéressent généralement à une ou deux étapes de la méthode globale que nous suivons. A titre indicatif, on peut trouver des exemples d'analyses descriptives dans [BAS 84, IYE 82, TRO 86]. Les tests de tendances sont étudiés dans [ASC 84, COX 66, GAU 90]. La définition et l'application des modèles de croissance de fiabilité sont discutées dans plusieurs publications et ouvrages tels que [MUS 87, XIE 91] par exemple. Par rapport à nos travaux antérieurs, des applications pratiques de toutes ou partie(s) des étapes de la méthode sont effectuées [KAA 90, KAN 91, KAN 87]. Le rôle des analyses de tendance est étudié plus en détail dans [KAN 94, KAN 95]. SoRel est présenté dans [KAN 92].

3. Filtrage, partitionnement et analyses descriptives des données

Les opérations relatives au filtrage, au partitionnement des données et aux analyses descriptives sont étroitement liées a) au système considéré et à la nature de l'application, b) à la collecte de données, c) aux fonctions du logiciel et à sa structure, d) aux objectifs de l'analyse, etc. Ces opérations ne sont pas nécessaires pour toute étude de fiabilité, par exemple la répartition des données en sous ensembles et les analyses descriptives peuvent ne pas être effectuées en fonction du niveau de détail recherché.

Le filtrage des données est nécessaire afin de ne garder que les données relatives au logiciel. Généralement les données collectées incluent également des rapports correspondant à des données non cohérentes, à des défaillances du matériel, à des données dupliquées (rapports liés à la même défaillance), etc. Ces rapports doivent être éliminés afin de ne garder que ceux concernant les fautes activées durant l'exécution du logiciel ayant conduit à défaillance et celles détectées durant l'analyse statique (relecture des spécifications, inspection de code). Les rapports dupliqués ne doivent pas être confondus avec les rapports relatifs à ce qui est communément appelé "redécouvertes" qui sont liés à la même faute (correspondant à des défaillances résultant de l'activation de la même faute). Ces derniers doivent être gardés car ils correspondent effectivement à des défaillances du logiciel distinctes ayant eu lieu sur la même installation ou sur des installations différentes : en fonction de leur nombre, elles peuvent influencer la fiabilité du système.

Le filtrage des données est généralement très fastidieux et peut prendre beaucoup de temps : en plus d'une bonne connaissance du système, du logiciel et de la collecte des données, il est parfois nécessaire d'interroger les personnes impliquées dans le test et la collecte. Le rapport entre le nombre de données retenues après filtrage et le nombre total de données (observé sur des systèmes que nous avons analysés [KAA 90] ou analysés par d'autres auteurs [BAS 84, LAR 94, LEV 90]) est d'environ 50%. Cette étape est essentielle car la qualité des résultats repose sur la qualité des données retenues ; elle doit être effectuée minutieusement.

A partir des rapports de défaillance retenus, on crée des fichiers représentant les données sous deux formes possibles : *temps jusqu'à défaillance* ou *nombre de défaillances par unité de temps*. Le choix entre une forme ou l'autre est

généralement guidé par l'objectif de l'étude (évaluation de la fiabilité en exploitation, suivi du développement et planification de la maintenance) et de la phase du cycle de vie concernée. L'utilisation des données sous la forme nombre de défaillances par unité de temps (ou intensité de défaillance) réduit l'impact des fluctuations locales sur la fiabilité du logiciel. Le choix de l'unité de temps dépend de l'utilisation du système et du nombre de défaillances observées sur la période concernée ; elle peut être différente pour des phases distinctes (dans ce cas, les données doivent impérativement être considérées séparément pour chaque phase).

Le partitionnement des données en sous ensembles permet d'effectuer des analyses plus complètes. Plusieurs partitionnements peuvent être effectués selon par exemple : la sévérité des défaillances, la localisation des fautes, la phase du cycle de vie ou le type d'installation (quand le logiciel est installé sur plusieurs systèmes avec des profils d'utilisation différents).

L'application des modèles de croissance de fiabilité aux défaillances les plus critiques permet d'évaluer le taux de défaillance relatif aux conséquences les plus sévères. La répartition en fonction de la localisation des fautes permet de mesurer l'influence de chaque composant (ou des composants les plus significatifs) sur le taux de défaillance global du logiciel. Ainsi l'accent sera mis sur les composants les moins fiables lors de la maintenance du logiciel ou lors de l'introduction de nouvelles versions.

Les analyses descriptives consistent à effectuer une synthèse des phénomènes observés sous forme de graphiques ou de corrélations afin d'identifier les tendances les plus significatives. Elles peuvent correspondre à des analyses simples telles que : la densité de fautes des différents composants ou la typologie des fautes, ou à des analyses combinées telles que la relation entre a) la localisation des fautes et leur sévérité ou b) les conditions d'occurrence des défaillances et leur sévérité, et c) la densité de fautes et la taille des composants. Ces analyses sont très utiles et les résultats correspondants sont souvent exploités par les compagnies qui ont un processus de développement contrôlé afin d'améliorer la qualité du logiciel [GRA 87], [ROS 89]. L'accumulation et l'analyse d'informations relatives à plusieurs projets, produits et versions permet d'avoir une meilleure connaissance du produit et d'analyser l'impact du processus de développement sur le produit.

4. Analyse de la tendance

Les variations de la fiabilité (croissance, décroissance ou stabilisation) peuvent être analysées grâce à l'application des tests de tendance. Ces tests sont de nature statistique et fournissent des indicateurs de fiabilité. La présentation des tests de tendance est suivie par une discussion sur leur applicabilité en temps réel et par l'examen de quelques exemples de résultats.

4.1. *Tests de tendance*

Il existe un certain nombre de tests de tendance dont l'application permet de déterminer si le logiciel est en croissance ou en décroissance de fiabilité. On peut citer trois tests graphiques très simples qui consistent à tracer l'évolution a) des

temps jusqu'à défaillance, b) du nombre cumulé de défaillances, et c) de l'intensité de défaillance (nombre de défaillances par unité de temps). La tendance est déduite empiriquement à partir de la lecture des courbes. Les tests analytiques permettent de quantifier cette tendance. Parmi ces tests, notre choix porte sur le test de Laplace [GAU 90, KAN 94]. En effet des études antérieures ont montré sa supériorité par rapport à d'autres tests tels que les tests de Kendall ou de Spearman [GAU 90]. Quand les données sont sous forme temps jusqu'à défaillances, le facteur de Laplace à l'instant T est donné par :

$$u(T) = \frac{\frac{1}{N(T)} \sum_{i=1}^{N(T)} \sum_{j=1}^{i} \theta_j - \frac{T}{2}}{T \sqrt{\frac{1}{12\, N(T)}}}$$

où θ_j est le temps jusqu'à occurrence de la défaillance j calculé depuis la relance du logiciel après la défaillance j-1, et N[T] est le nombre total de défaillances observées dans [0,T]. Quand les données sont sous la forme intensité de défaillance (avec n(i), le nombre de défaillances pendant l'unité de temps i), l'expression de ce facteur est :

u(k) = **Erreur !**

L'utilisation pratique qui en est faite dans le contexte de la croissance de fiabilité est la suivante :

- les valeurs négatives indiquent une intensité de défaillance décroissante (croissance de fiabilité),
- les valeurs positives suggèrent une intensité de défaillance croissante (décroissance de fiabilité),
- les valeurs oscillant entre -2 et +2 indiquent une fiabilité stabilisée (la fiabilité n'est ni croissante ni décroissante).

Ces valeurs pratiques sont déduites des niveaux de signification associés au test statistique d'une distribution Normale. A titre indicatif, pour un niveau de signification de 5% : $|u(k)| < 1{,}96$ indique une fiabilité stabilisée.

4.2. *Tendance locale et globale*

A l'origine, le test de Laplace a été développé pour identifier la tendance globale sur un intervalle de temps ainsi que les changements globaux de la tendance (points où le facteur s'annule). La relation entre le facteur de subadditivité et le facteur de Laplace que nous avons montrée dans [KAN 94] a révélé que ce dernier est également capable de détecter les points de changements locaux de tendance : ils sont déterminés par les points de changement de tendance de la courbe représentant le facteur de Laplace (point T_{L1} et T_{L2} de la figure 2-a qui donne un exemple d'évolution de ce facteur). Si on effectue un changement de l'origine des temps à partir de T_{L1}, ce facteur devient négatif à partir de T_{L1} comme indiqué sur la figure 2-b. Notons qu'un changement de l'origine du temps n'entraîne pas une simple translation de la courbe (due à la présence de la variable temps dans le dénominateur) : la non prise en compte des données entre 0 et T_{L1} renforce les

variations locales et amplifie les variations du facteur de Laplace, sans déplacer toutefois les points de changements locaux de tendance. Ceci sera illustré sur des données réelles dans la partie 6.

De façon pratique l'évolution du facteur de Laplace peut être interprétée comme suit :

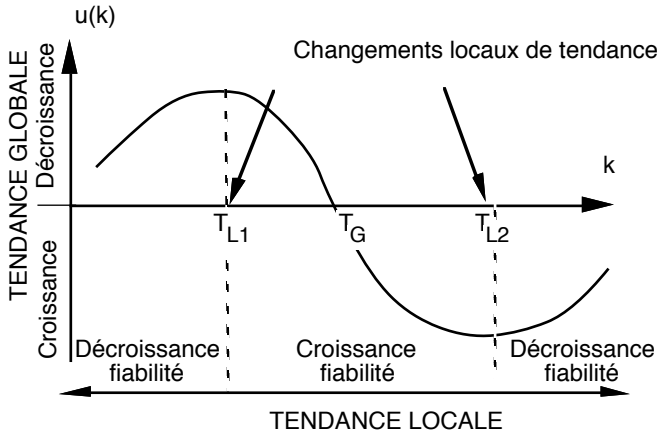
• les valeurs négatives (respectivement positives) du facteur de Laplace sur un intervalle de temps indiquent une croissance (respectivement décroissance de fiabilité) globale sur cet intervalle,

• les valeurs décroissantes (respectivement croissantes) sur un sous intervalle indiquent une croissance de fiabilité (respectivement une décroissance de fiabilité) locale sur ce sous intervalle.

La figure 3 résume quelques situations typiques de l'évolution du facteur de Laplace et montre le lien entre le nombre cumulé de défaillances, l'intensité de défaillance et le facteur de Laplace. Dans la pratique, nous utilisons le facteur de Laplace comme indicateur de tendance, considérant à la fois son signe et son sens de variation car il permet d'identifier aisément à la fois les tendances locale et globale.

4.3. *Comment utiliser les résultats de l'analyse de tendance ?*

Le rôle des tests de tendance est d'attirer l'attention sur des problèmes ou anomalies qui, sans leur utilisation, pourraient être ressentis beaucoup plus tard, voire trop tard, ce qui permet de trouver des solutions à ces problèmes assez rapidement. Il est clair que ces tests ne donnent pas directement la solution au problème et ne peuvent en aucun cas remplacer l'interprétation des personnes en charge du développement ou de l'exploitation. Des exemples d'utilisation des résultats de tests de tendance sont donnés dans ce qui suit.



-a-

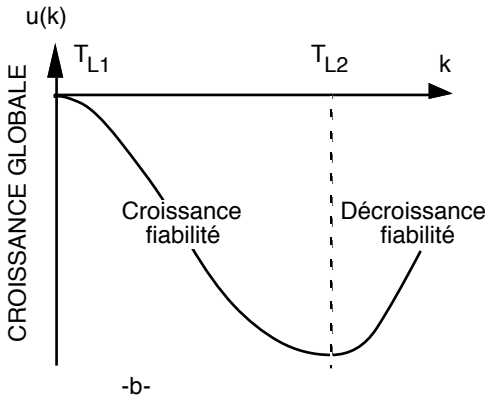


Figure 2. *Facteur de Laplace et changements de tendance*

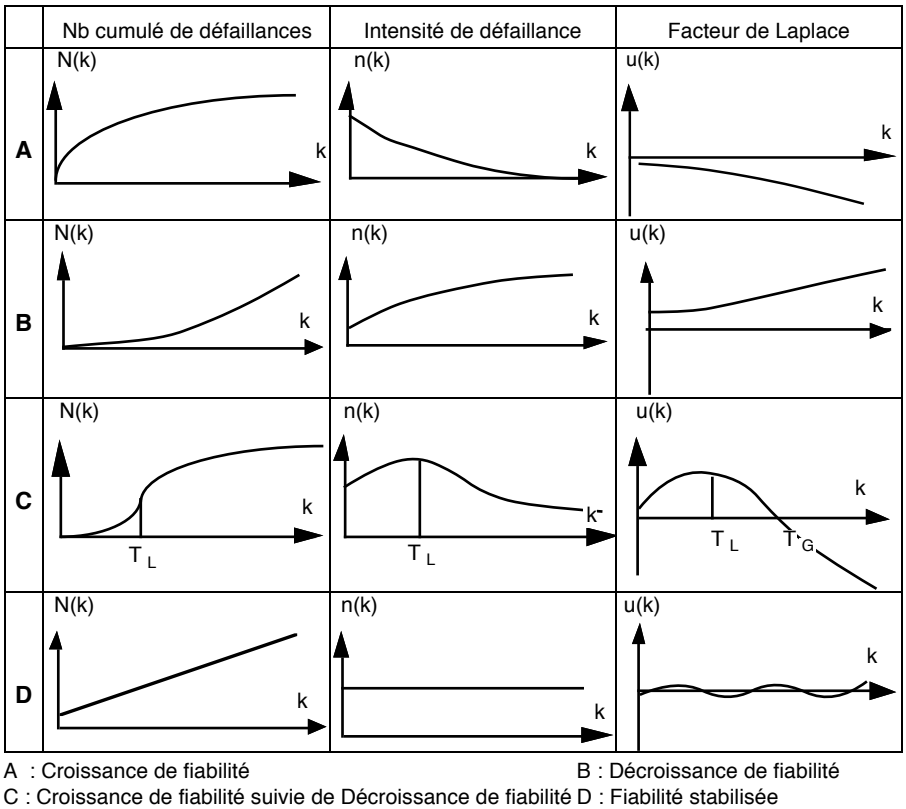


Figure 3. *Lien entre différents indicateurs de tendance*

Une *décroissance de fiabilité* au début d'une nouvelle activité telle que a) une nouvelle phase du cycle de vie, b) changement des jeux de test à l'intérieur d'une

même phase ou c) ajout de nouveaux utilisateurs, ou d) activation du système avec un profil d'utilisation différent, etc., est généralement attendue et peut être considérée comme normale si elle ne dure pas trop longtemps (cf. [KEN 92] par exemple). La décroissance de fiabilité peut aussi résulter de fautes de régression. La recherche de la cause de cette décroissance pourrait éventuellement révéler des lacunes dans le processus de développement et aider à réorienter le cas échéant l'activité de test. Une telle analyse peut aussi conduire à décider de réécrire la partie du logiciel dont la fiabilité ne cesse de décroître (le nombre de fautes activées par unité de temps ne fait que croître).

Une *croissance de fiabilité* faisant suite à une décroissance de fiabilité indique qu'après correction des premières fautes, l'activité de test correspondante révèle de moins en moins de fautes. Une croissance de fiabilité brutale et de durée relativement longue peut aussi résulter d'un ralentissement de l'activité de test ; elle peut aussi être due au fait que les défaillances ne sont pas enregistrées sur cette période de temps. Il est recommandé de bien analyser les raisons de ces changements.

Une *fiabilité stabilisée* avec très peu de défaillances indique que l'activité en cours a atteint ses limites : l'application des tests correspondants ne révèle plus que peu de fautes et les corrections introduites n'ont plus d'effets perceptibles. Ceci suggère de passer à la phase suivante ou de livrer le logiciel si on estime que le test arrive à sa fin. De façon générale un ensemble de tests doit être appliqué aussi longtemps qu'il continue à entraîner une croissance de fiabilité et peut être arrêté à partir du moment où une fiabilité stabilisée est atteinte. Le fait que la fiabilité stabilisée ne soit pas atteinte pourrait contribuer à la décision de continuer à tester le logiciel plutôt que de le livrer.

Notons enfin que l'analyse de la tendance permet aussi de guider le choix d'un modèle de croissance de fiabilité afin d'évaluer les mesures recherchées comme nous le verrons dans la partie suivante.

5. Modèles de croissance de fiabilité

Un grand nombre de modèles a été développé depuis une trentaine d'années[2], cependant leur utilisation pratique a révélé qu'aucun modèle ne donne des résultats meilleurs que les autres pour tout ensemble de données collectées et dans n'importe quelles conditions. L'un des objectifs de notre approche est d'aider ces modèles à donner de meilleurs résultats.

Dans cette partie nous analysons la pertinence des résultats issus des modèles de croissance de fiabilité en fonction de la phase du cycle de vie et montrons comment on peut utiliser conjointement les tests de tendance et les modèles de croissance de fiabilité. Toutefois, afin de faciliter la compréhension des résultats de l'application des modèles de croissance de fiabilité aux données réelles traitées dans la partie 6, nous donnons à la figure 4 l'expression de l'intensité de défaillance ou du taux de défaillance associés aux modèles qui seront appliqués à ces données.

---

[2] Le lecteur peut trouver plus de détails sur les modèles de croissance de fiabilité et les problèmes liés à leur mise en œuvre pratique dans les références [MUS 87] ou [XIE 91].

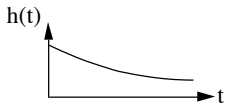
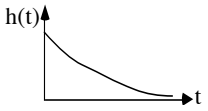
Modèle	Allure de h(t) ou λ(t)
Hyperexponentiel (HE) [LAP 91] $h(t) = \frac{\omega\zeta_{sup}e^{-\zeta_{sup}t} + \varpi\zeta_{inf}e^{-\zeta_{inf}t}}{\omega e^{-\zeta_{sup}t} + \varpi e^{-\zeta_{inf}t}}$	
Exponentiel (EXP) [GOE 79] $h(t) = N\,\phi\,\exp^{-\phi t}$	
Forme de S (SS) [OHB 84] $h(t) = N\,\phi^2\,t\,\exp^{-\phi t}$	

Figure 4. *Modèles utilisés par la suite*

5.1. *Pertinence des résultats des modèles de croissance de fiabilité*

L'application des modèles de croissance de fiabilité aux données collectées sur un système permet d'évaluer les mesures de fiabilité telles que le taux de défaillance, l'intensité de défaillance ou le nombre cumulé de défaillances. La pertinence des résultats varie considérablement avec la phase du cycle de vie considérée :

• Durant les premières phase du développement, les résultats ne sont généralement pas significatifs : les temps jusqu'à défaillance observés sont de l'ordre de quelques minutes à quelques heures, les estimations obtenues ne peuvent être que du même ordre de grandeur ou supérieures d'un ordre de grandeur ; elles sont donc (fort heureusement) loin de ce qui est espéré en vie opérationnelle. De plus, bien souvent, le profil d'exécution est différent du profil opérationnel. Dans ce cas, la validation du logiciel peut être guidée par l'analyse de la tendance comme nous l'avons vu dans la partie précédente. Les estimations peuvent néanmoins servir pour estimer l'effort de test nécessaire pour effectuer les corrections correspondantes. Cependant, étant donné la fréquence de changement des conditions d'utilisation, on ne peut espérer que des estimations à court terme.

• Durant les dernières phases de la validation, quand le logiciel devient plus fiable, les temps jusqu'à défaillance deviennent plus grands et les résultats des modèles plus convaincants, plus particulièrement quand le logiciel est activé selon un profil d'exécution proche du profil opérationnel. Cependant, un très petit nombre de défaillances peut rendre les données de défaillance inexploitables par les modèles de croissance de fiabilité.

• En exploitation, quand le logiciel est installé sur plusieurs systèmes, les résultats sont généralement très significatifs car nous disposons de véritables données statistiques (voir par exemple [ADA 84] pour des systèmes d'exploitation, [KAN

87] pour des autocommutateurs téléphoniques ou [LAR 94] pour des applications nucléaires).

5.2. *Modèles de croissance de fiabilité et tendance*

L'application des modèles de fiabilité sans traitement préalable peut mener à des résultats non réalistes quand les données collectées suivent une tendance différente de celle supposée par les modèles. A l'inverse, quand les modèles sont appliqués à des données ayant la même tendance que celle correspondant à leurs hypothèses, les résultats sont considérablement améliorés [KAN 87, KAN 93]. Du fait que les modèles existants ne permettent de modéliser que deux types de situations : a) intensité de défaillance décroissante (croissance de fiabilité) et b) intensité croissante puis décroissante (décroissance suivie de croissance), il est essentiel d'appliquer les modèles de croissance de fiabilité sur des sous-ensembles de données pour lesquels la tendance est en accord avec les hypothèses de modélisation. Ces sous-ensembles sont identifiés grâce à l'analyse de tendance. En fait les modèles de croissance de fiabilité peuvent être appliqués de la façon suivante :

- zone de croissance de fiabilité (figure 3, situation A) : n'importe quel modèle modélisant la croissance de fiabilité uniquement peut être appliqué,
- zone de décroissance suivie de croissance (figure 3, situation C) : un modèle en forme de S (S-Shaped) peut être appliqué,
- zone de fiabilité stabilisée (figure 3, situation D), un modèle à taux constant est suffisant (processus de Poisson homogène).

5.3. *Modèles de croissance de fiabilité et tendance en temps réel*

Les modèles peuvent être appliqués de façon rétrospective (l'objectif est d'évaluer les mesures de fiabilité sur la période de temps passée à partir des données collectées) ou de façon prévisionnelle (l'objectif est alors de prévoir le comportement du logiciel pour une période future). Dans ce dernier cas, les prévisions sont aussi basées sur les observations effectuées sur le système et on ne peut se servir que des informations disponibles au moment de l'évaluation. Il est donc très important de s'assurer de la validité des résultats ou du moins de s'assurer que les résultats ne sont pas aberrants sur la période d'observation (validité rétrospective).

L'analyse de la tendance permet dans ce cas de bien choisir la période de temps passée sur la quelle le modèle sera appliqué ainsi que le modèle à appliquer. En général, on peut avoir confiance dans les résultats d'un modèle aussi longtemps que les conditions d'utilisation du système restent globalement inchangées (pas de changements majeurs dans la stratégie de test ou dans les spécifications, pas d'introduction de nouveaux utilisateurs avec des profils d'utilisation significativement différents des autres…). Néanmoins même dans ces situations, une décroissance de fiabilité peut avoir lieu. Dans un premier temps, on peut considérer qu'elle résulte de fluctuations locales dues aux aléas des occurrences des défaillances, et qu'elle va disparaître prochainement ; on continue donc à avoir confiance dans les résultats des évaluations. Si la fiabilité continue à décroître de façon significative, on ne peut plus faire confiance aux résultats des prévisions ; il

faut attendre qu'il y ait à nouveau croissance de fiabilité, faire une nouvelle partition des données et effectuer de nouvelles prévisions.

Si les conditions d'utilisation changent, une attention toute particulière doit être portée aux changements de tendance qui peuvent en résulter et qui peuvent entraîner des prévisions erronées. L'application des tests de tendance est d'une grande aide :

- si aucun changement de tendance n'est perçu, on continue à avoir confiance dans les résultats,
- si, au contraire, une décroissance de fiabilité se fait sentir, il faut attendre qu'il y ait à nouveau croissance de fiabilité, faire une partition des données et effectuer de nouvelles prévisions.

6. Étude d'un cas réel

Afin d'illustrer certains aspects de la méthode, nous considérons les données collectées sur le logiciel d'un autocommutateur téléphonique observé pendant une période couvrant la fin de la validation et le début de la vie opérationnelle. L'autocommutateur permet le raccordement de 4096 abonnés. Le logiciel a été développé selon un processus développement en cascade (“Waterfall model”) ; il est écrit en langage Assembleur. Les données relatives à l'activation et à l'élimination des fautes dans le logiciel ont été répertoriées dans des rapports de défaillance (RD). Le filtrage des informations contenues dans ces rapports a permis de retenir les seuls rapports qui sont utiles pour l'étude de la fiabilité du logiciel. Pendant la période considérée (32 unités de temps), 210 RDs ont été retenus : 73 RDs pendant la fin de la validation (8 unités de temps) et 137 RDs pendant le début de la vie opérationnelle (24 unités de temps). Il est à noter que le nombre de systèmes (autocommutateurs) mis en service durant la vie opérationnelle à augmenté progressivement et a atteint le nombre de 42 à la fin de la période considérée.

Nous ne pouvons malheureusement pas, par manque de place, illustrer toutes les étapes de la méthode que nous avons présentées dans les parties précédentes. Nous nous limitons à l'analyse de la tendance appliquée à toutes les données collectées et l'évaluation de la fiabilité du logiciel en vie opérationnelle. Outre l'évaluation de la fiabilité opérationnelle du logiciel et de ses composants, nous évaluons les mesures relatives à la prévision de l'effort de maintenance nécessaire pour la correction des fautes du logiciel.

6.1. *Décomposition du logiciel*

Le logiciel est structuré en modules élémentaires. Ces modules peuvent être regroupés en quatre composants correspondant aux principales fonctions mises en œuvre :

- **Téléphonie** : traitement local d'appels et taxation.
- **Défense** : tests en ligne, mesures de trafic, supervision des alarmes.
- **Interface—périphériques** : communication et gestion des interfaces avec les périphériques locaux (mémoires de masse, terminaux et panneaux d'alarmes).

• **Interface—joncteurs** : gestion des signaux acoustiques et des échanges avec les organes extérieurs (contrôle des joncteurs d'entrée/sortie, et des canaux numériques).

Les données collectées peuvent être ainsi regroupées en sous ensembles de données conformément à cette décomposition. La figure 5 donne la taille (en kilo-octets) et le nombre de corrections effectuées sur chacun de ces composants durant la vie opérationnelle. Il à noter que la somme des corrections effectuées sur l'ensemble des composants (146) est supérieure au nombre de corrections effectuées sur le logiciel global durant la vie opérationnelle (137). Ceci s'explique par le fait que 11 RDs ont conduit à la correction de plusieurs composants à la fois illustrant ainsi la non indépendance totale des différents composants vis-à-vis de l'occurrence des défaillances. Cependant, il faut noter que cette dépendance est relativement faible (environ 10% des RDs correspondent à ce cas).

	Taille (kilo-octets)	Nombre de corrections
Téléphonie	75	40
Défense	103	47
Interface—périphériques	115	41
Interface—joncteurs	42	18
Total	335	146

Figure 5. *Taille des composants et nombre de corrections effectuées en opération*

6.2. *Analyse de la tendance*

La figure 6-a donne les valeurs du facteur de Laplace calculées à partir de toutes les données collectées et en considérant l'ensemble des systèmes mis en service pour ce qui concerne la vie opérationnelle. On peut remarquer qu'il y a eu une décroissance de fiabilité entre les unités de temps 4 et 6. Cette décroissance résulte du changement du type de tests appliqués pendant cette période de la validation conduisant à l'activation de nouvelles parties du logiciel. On peut noter également que le logiciel a été installé sur les sites opérationnels avant d'observer une croissance significative de sa fiabilité. Cependant, on peut remarquer qu'il y a eu globalement une tendance de croissance de fiabilité durant la vie opérationnelle. La figure 6-b montre l'évolution du facteur de Laplace en considérant les données relevées pendant la vie opérationnelle uniquement, permettant ainsi de révéler des fluctuations locales de la tendance durant la période couvrant les unités de temps 14 à 24. Ces fluctuations sont difficiles à percevoir à partir de la figure 6. Ces résultats confirment ainsi que l'élimination d'une partie des données qui sont à l'origine d'une décroissance de fiabilité (c'est-à-dire correspondant à une valeur positive du facteur de Laplace) a) conduit à des valeurs négatives du facteur de Laplace et b) amplifie les fluctuations locales de la tendance (cf. §. 4-2). Les fluctuations relatives à la période 14-24 sont dues à la mise en service progressive de 40 nouveaux systèmes durant cette période.

Afin d'analyser l'évolution de la fiabilité telle qu'elle est perçue par un système moyen, on peut considérer les intensités de défaillance divisées par le nombre de

systèmes en service et appliquer le test de Laplace aux données obtenues. Les résultats sont donnés sur la figure 7 : la courbe obtenue est plus lisse que celle de la figure 6-b et les fluctuations locales ont disparu. Notons que les variations observées entre les unités de temps 14 et 17 sont dues à la mise en service de 6 nouveaux systèmes.

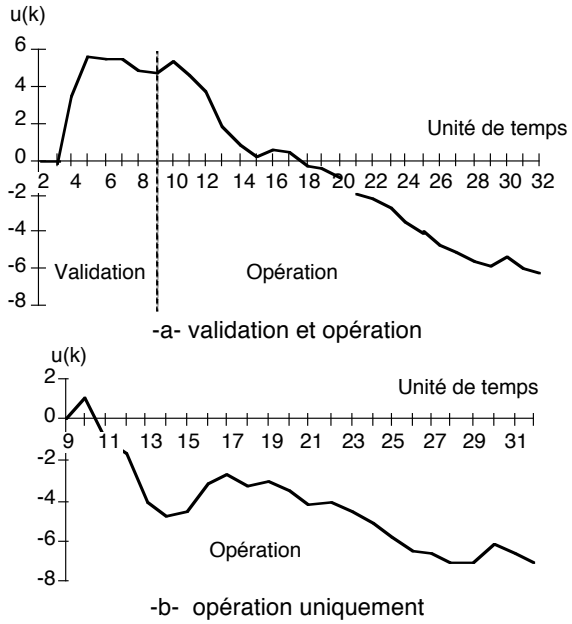


Figure 6. *Facteurs de Laplace pour tous les systèmes*

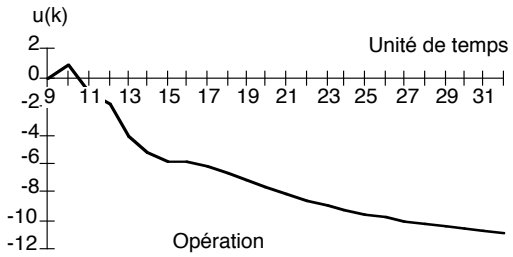


Figure 7. *Facteur de Laplace pour un système moyen*

On peut noter à partir de la figure 7 que la croissance de fiabilité du logiciel est maintenue pendant la vie opérationnelle et tend à être monotone à partir de l'unité de temps 17. Ce comportement a été observé également pour les quatre composants du logiciel (en considérant un système moyen aussi) comme l'illustre la figure 8. Cette figure révèle aussi des fluctuations locales de tendance pour tous les composants durant la période 14-16, et surtout pour le composant “Interface—joncteurs”

6.3. *Application de modèles de croissance de fiabilité*

Nous adoptons d'abord le point de vue de l'utilisateur en évaluant l'intensité de défaillance et le taux de défaillance résiduel du logiciel en opération à partir des données relatives à un système moyen. Ensuite, nous adoptons le point de vue du fournisseur en évaluant le nombre moyen de défaillances estimé pour une période future et pour l'ensemble des systèmes en service, dans un objectif de planification de l'effort de maintenance.

6.3.1. *Taux de défaillance résiduels du logiciel global et de ses composants*

Les résultats de l'analyse de tendance présentés dans la figure 7 montrent une croissance de fiabilité monotone durant les 15 dernières unités de temps. Ces résultats permettent de guider l'application des modèles de croissance de fiabilité ; ils suggèrent le choix d'un modèle qui décrit une croissance de fiabilité monotone (c'est-à-dire à intensité de défaillance décroissante). L'intensité de défaillance et le taux de défaillance résiduel du logiciel sont évalués à partir du modèle hyperexponentiel (HE) qui est le seul modèle qui permet d'évaluer cette mesure. Le modèle HE est appliqué au logiciel global et à ses composants durant les 15 dernières unités de temps. Les intensités de défaillance estimées par le modèle HE pour chacun des composants et les intensités de défaillance observées sont présentées sur la figure 9. La figure 10 donne les valeurs des taux de défaillance résiduels correspondants estimés par HE : Interface—périphériques est le composant le moins fiable et Téléphonie est le composant le plus fiable.

Afin d'apprécier la validité des résultats fournis par le modèle, nous présentons également sur la figure 10 les valeurs moyennes du taux de défaillance calculées directement à partir des données observées durant les 10 dernières unités de temps. On peut remarquer que les valeurs observées sont légèrement supérieures aux valeurs estimées par le modèle, ce qui est normal étant donné que le modèle estime le comportement du logiciel quand il atteint un régime stabilisé. Cependant, on constate que les deux valeurs sont très proches pour les composants Défense et Interface—périphériques ; on peut en conclure que ces deux composants ont pratiquement atteint leur régime stabilisé pendant les 10 dernières unités de temps alors que la fiabilité des composants Téléphonie et Interface—périphériques a continué à évoluer pendant cette période.

On peut noter que les différents composants ont des comportements différents : l'amélioration de la fiabilité des composants “Téléphonie” et “Interfaces—joncteurs” est plus rapide que celle des deux autres composants ; de plus ces composants possèdent relativement les intensités de défaillance les plus élevées pendant les premières unités de temps de la période considérée. Les composants “Défense” et “Interface-périphériques” sont caractérisés par des intensités de défaillance plus faibles (c'est-à-dire une meilleure fiabilité) au début ; cependant l'amélioration de leur fiabilité a été plus lente pendant les périodes suivantes ce qui explique le fait que vers la fin de cette période ils ont été moins fiables que les deux autres composants.

La figure 11 présente les résultats fournis par l'application directe du modèle hyperexponentiel aux données relatives au logiciel global.

Les figures 9 et 11 suscitent les commentaires suivants :

• L'évaluation de l'intensité de défaillance du logiciel global uniquement ne permet pas d'observer l'évolution des intensités de défaillance des composants et masque de ce fait cette évolution.

• Le taux de défaillance résiduel estimé à partir de l'application directe du modèle HE à toutes les données ($5{,}7\ 10^{-5}$/h) est du même ordre de grandeur que celui obtenu par la sommation des intensités résiduelles des composants estimées ($5{,}3\ 10^{-5}$/h) bien que les composants ne soient pas totalement stochastiquement indépendants vis-à-vis de l'occurrence des défaillances.

• On peut penser que le taux de défaillance résiduel estimé pour le logiciel étudié est relativement élevé si on le compare à d'autres logiciels de télécommunication (de l'ordre de 10^{-6}/h). Cependant, il faut remarquer que toutes les défaillances observées du logiciel ont été considérées dans cette analyse, quelle que soit leur sévérité. Le taux estimé inclut donc aussi bien des défaillances dont les conséquences sont mineures que des défaillances ayant des conséquences plus graves. Malheureusement, les informations concernant la sévérité des défaillances n'ont pas été systématiquement incluses dans les relevés de défaillance ce qui nous a empêché d'évaluer la fiabilité du logiciel en tenant compte des conséquences des défaillances.

6.3.2. *Planification de l'effort de maintenance*

Nous adoptons dans ce paragraphe le point de vue du fournisseur (plus précisément celui du responsable de la maintenance) dans un objectif de planification de l'effort de maintenance nécessaire pour la correction des fautes résiduelles qui pourraient être activées pendant la vie opérationnelle. Dans cet objectif, nous considérons toutes les défaillances relevées à partir de tous les systèmes en service; les données considérées sont celles de la figure 6-b.

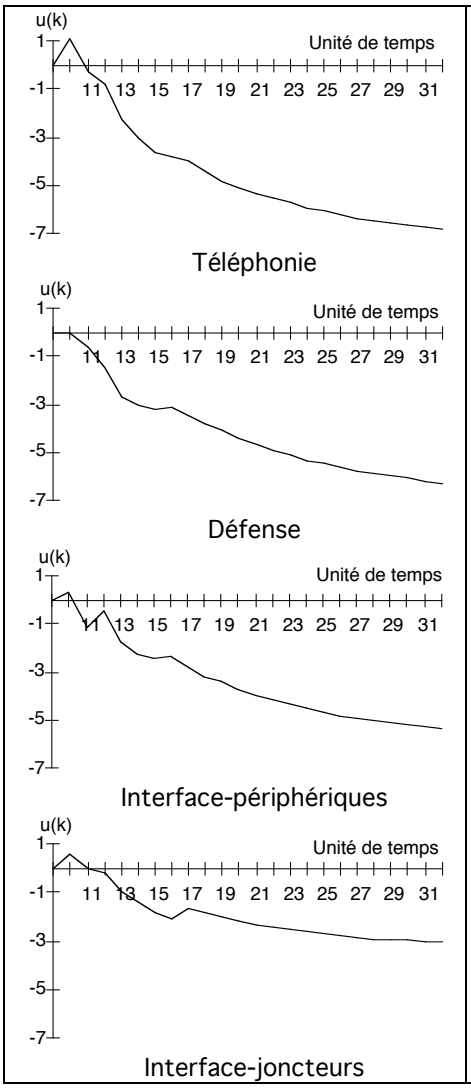


Figure 8. *Facteurs de Laplace*

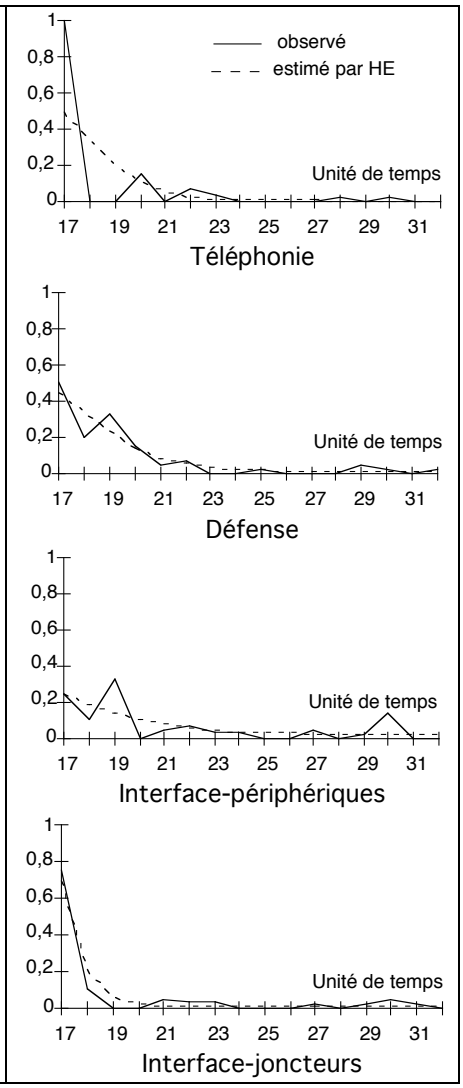


Figure 9. *Intensités de défaillance*

	Taux résiduel estimé par HE	Taux moyen observé (10 dernières unités de temps)
Téléphonie	1,2 10^{-6}/h	1,0 10^{-5}/h
Défense	1,4 10^{-5}/h	1,6 10^{-5}/h
Interface—périphériques	2,9 10^{-5}/h	3,7 10^{-5}/h
Interface—joncteurs	8,5 10^{-6}/h	2,0 10^{-6}/h

Figure 10. *Taux de défaillance moyens observés et résiduels estimés par HE*

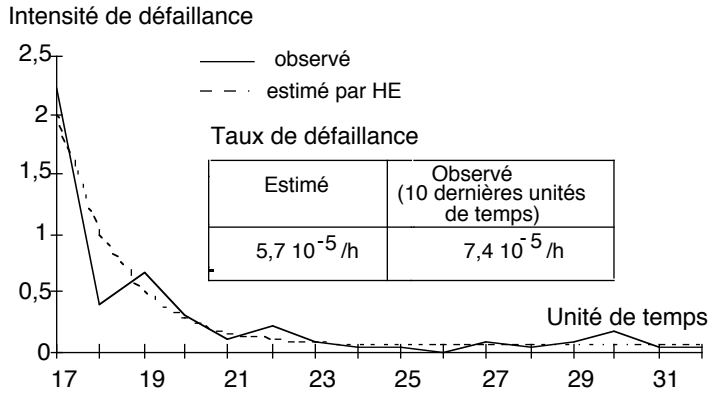
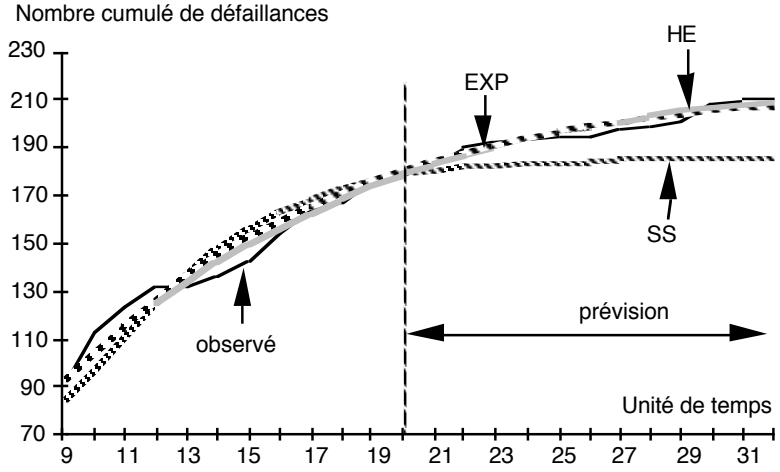


Figure 11. *Intensités de défaillance observées et estimées par HE*

La figure 6-b montre que le logiciel a manifesté une croissance de fiabilité globale sur toute la période opérationnelle. Ceci suggère le choix d'un modèle de croissance de fiabilité qui modélise une intensité de défaillance décroissante : les modèles HE et le modèle exponentiel (EXP) appartiennent à cette catégorie. La figure 12 présente les résultats d'application de ces modèles en utilisant les données correspondant à la vie opérationnelle. En plus de ces deux modèles, nous avons appliqué le modèle en forme de S (SS) afin de comparer les résultats des différents modèles et d'illustrer l'intérêt de faire une analyse de tendance avant toute application de modèle.

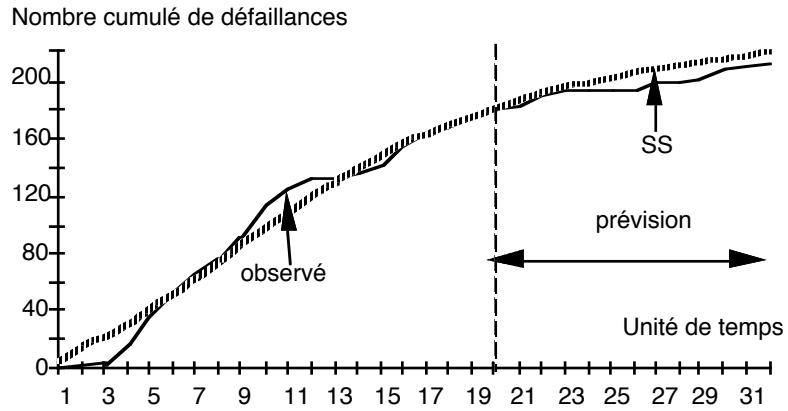


Nb défaillances	EXP	HE	SS	Observées
de 20 à 32	33	37	9	34

Figure 12. *Nombre cumulé de défaillances en opération estimé à partir de [9 - 20]*

Pour obtenir les résultats de la figure 12, nous avons utilisé les données correspondant à la moitié de la période considérée (unités de temps 9 à 19) pour estimer les paramètres des modèles, et évaluer de façon prévisionnelle le nombre cumulé de défaillances qui pourraient être observées durant les périodes suivantes. La figure 12 montre, de façon non surprenante, que les modèles HE et EXP fournissent des résultats qui sont proches des valeurs observées alors que les résultats de SS sont trop optimistes. Les prévisions du nombre cumulé de défaillances pendant la période 20-32 sont égales à 37 et 33, pour HE et EXP respectivement, et uniquement 9 pour SS ; ces valeurs sont à comparer avec 34 qui est le nombre de défaillances observées pendant cette période.

Les mauvais résultats obtenus à partir du modèle SS ne sont pas surprenants compte tenu du fait que, durant la période considérée, les hypothèses de ce modèle ne sont pas satisfaites. En effet, ce modèle modélise une intensité de défaillance qui est initialement croissante (décroissance de fiabilité) et qui décroît par la suite (croissance de fiabilité). La figure 6-a suggère l'application du modèle SS en utilisant les données collectées dès le début de la période d'observation afin d'intégrer le changement de tendance qui a eu lieu pendant la période couvrant les unités de temps 5 à 9 afin de satisfaire les hypothèses du modèle. Les résultats sont donnés sur la figure 13 et on peut observer une nette amélioration des prévisions fournies par le modèle.



Nb défaillances	SS	Observées
de 20 à 32	40	34

Figure 13. *Nombre cumulé de défaillances estimé par SS à partir de [1 -20]*

L'exemple ci-dessus montre que l'utilisation des tests de tendance peut améliorer de façon significative les résultats des modèles si ceux-ci sont appliqués à des données qui représentent un comportement cohérent avec leurs hypothèses. De plus, si ce critère est respecté, on peut appliquer plusieurs modèles en considérant des périodes de temps différentes et obtenir des résultats équivalents.

7. SoRel : outil d'analyse et d'évaluation de la fiabilité du logiciel

Afin de faciliter la mise en œuvre de la méthode, nous avons développé SoRel qui est un outil d'aide à l'analyse de la tendance et à l'évaluation de la fiabilité du logiciel. Il est constitué de deux modules permettant a) l'exécution de tests de tendance (TENDANCE), et b) l'application de modèles de croissance de fiabilité (MODÈLES). Les données traitées par SoRel peuvent être soit sous la forme temps jusqu'à défaillances, soit sous la forme nombre de défaillances par unité de temps.

SoRel est mis en œuvre sur un Macintosh II-xx possédant un co-processeur arithmétique. Il est doté d'une interface homme-machine conviviale utilisant en particulier les facilités de multi-fenêtrage et d'interactivité du Macintosh. Le programme propose à l'utilisateur un ensemble de tests de tendance et de modèles de croissance de fiabilité. Néanmoins, SoRel est développé de façon modulaire ce qui permet d'introduire facilement de nouveaux tests de tendance ou modèles de croissance sans modifier de façon significative la structure du programme. Le programme est écrit en Pascal (5 000 lignes de code source) ; le code objet occupe environ 250 kilo-octets.

Les deux modules (TENDANCE et MODÈLES) utilisent les mêmes fichiers en entrée qui peuvent être créés et modifiés par n'importe quel logiciel de traitement de texte ou éditeur graphique. Les résultats numériques d'exécution sont affichés directement sur l'écran pendant l'exécution et peuvent être représentés, sur demande de l'utilisateur, sous forme de courbes. Ces résultats sont sauvegardés sous la forme de fichiers ASCII et peuvent être utilisés par d'autres applications Macintosh (Excel en particulier) afin de comparer les résultats obtenus à partir de différentes applications de modèle par exemple.

7.1. *Le module "TENDANCE"*

SoRel permet d'exécuter plusieurs tests de tendance. Le choix d'un test de tendance en fonction du type de données de défaillance est effectué à l'aide de fenêtres sous forme de menus ou d'options à sélectionner comme illustré sur les figures 14 et 15. Le choix d'un fichier de données en entrée sur un disque peut être effectué de façon automatique en parcourant l'arborescence du disque (cf. figure 16) : il n'est pas indispensable que le fichier d'entrée soit dans le même dossier que celui de l'application.

Les tests de tendance peuvent être appliqués à toutes les données du fichier d'entrée ou bien uniquement à un sous-ensemble. SoRel propose aux utilisateurs de choisir de façon interactive, par le biais d'une fenêtre affichée à l'écran, le sous-ensemble des données auquel le test doit être appliqué (cf. figure 17). La figure 18 présente un exemple de résultats graphiques fournis par "TENDANCE". Les résultats sont stockés sous forme de fichiers et peuvent être utilisés par d'autres applications.

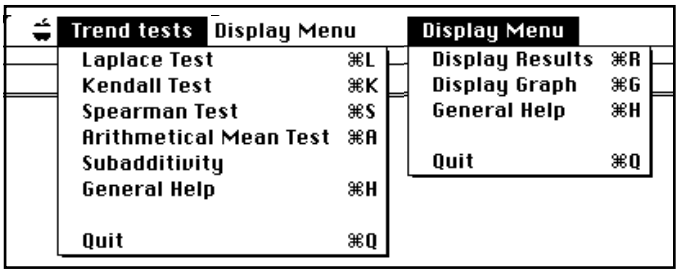


Figure 14. *Choix d'un test de tendance*

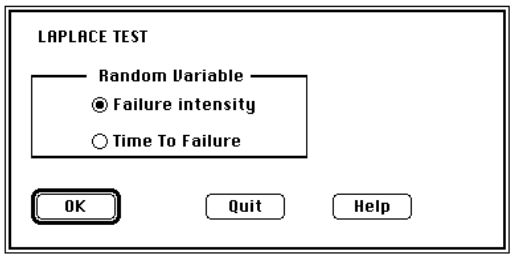


Figure 15. *Choix du type de données*

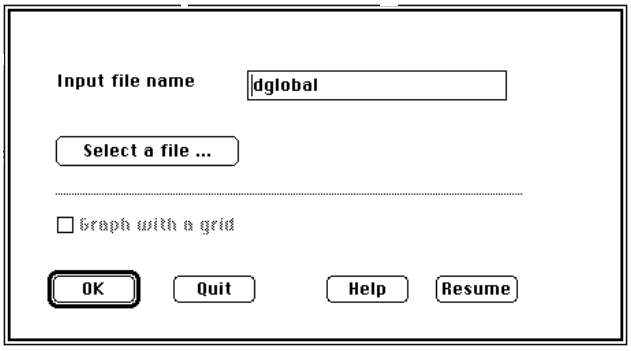


Figure 16. Sélection du fichier d'entrée

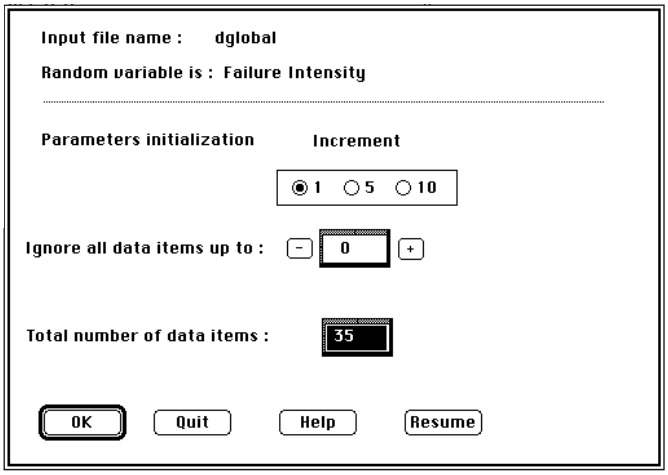


Figure 17. Définition du sous-ensemble des données d'entrée

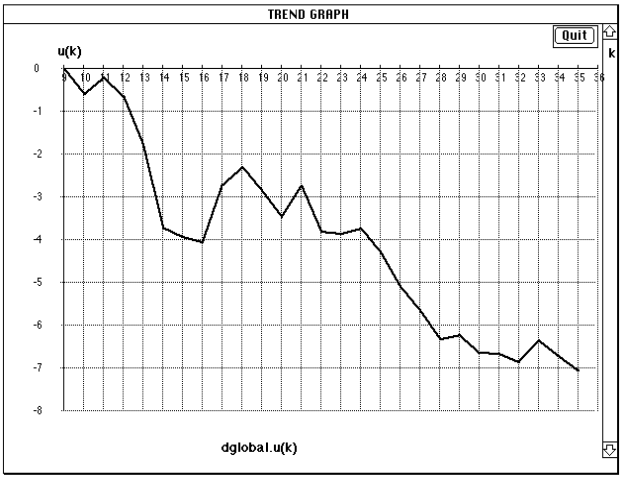


Figure 18. *Résultats graphiques fournis par "TENDANCE"*

7.2. Le module "MODÈLE"

L'utilisateur sélectionne le modèle (Figure 19) le mieux adapté aux sous-ensembles de données selon le résultat des tests de tendance et également en fonction des mesures qui correspondent aux objectifs de l'étude de fiabilité. Il doit indiquer le type des données d'entrée ainsi que la mesure qu'il cherche à évaluer (cf. figure 20), il doit ensuite fournir un certain nombre d'éléments d'information en fonction du modèle choisi (cf. figure 21).

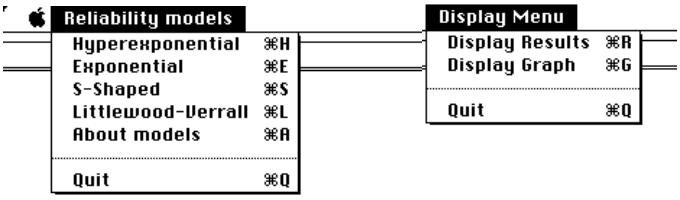


Figure 19. *Choix du modèle*

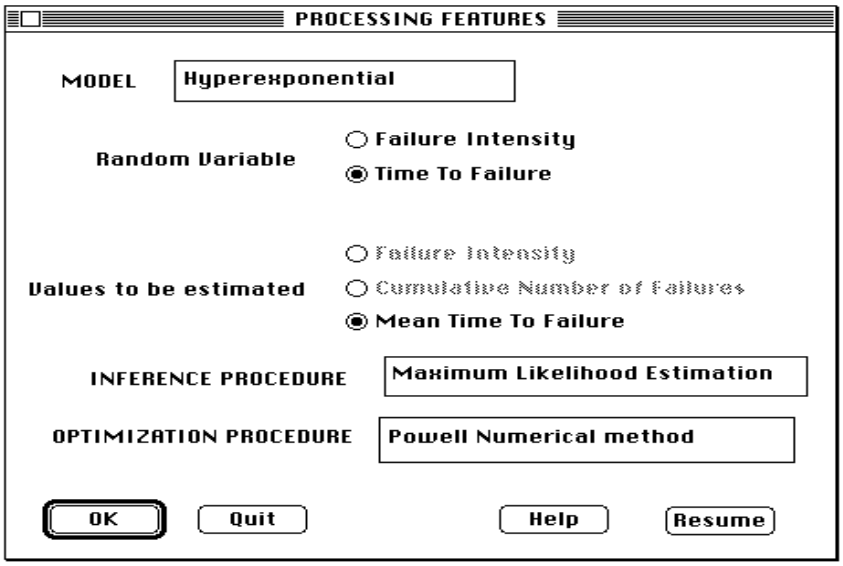


Figure 20. *Choix du type des données d'entrée et de la mesure à évaluer*

La zone **PS**, destinée à l'initialisation des paramètres du modèle, est nécessaire quand l'estimation de ces paramètres est effectuée par la méthode d'optimisation numérique de Powell. L'utilisateur doit fournir dans ce cas des valeurs initiales des paramètres qui se rapprochent de l'optimum. La zone **DP**, destinée au partitionnement des données, permet à l'utilisateur a) de choisir le sous-ensemble de données qui servira à l'estimation des paramètres du modèle, et b) l'intervalle de temps pour lequel il cherche à obtenir des prévisions. Le programme laisse également à l'utilisateur le choix d'utiliser ou non une fenêtre glissante pour calibrer le modèle (zone **C**). Dans le cas où cette option est validée, l'utilisateur doit fixer la taille de la fenêtre souhaitée ou bien utiliser la valeur par défaut qui est égale à 20. Dans ce cas, l'estimation des paramètres est basée sur les 20 dernières données du fichier d'entrée et non pas sur toutes les données comme c'est le cas lors d'une utilisation classique sans fenêtre. Enfin, la zone **V**, permet de fixer la période pour laquelle les critères de validation des résultats du modèle sont évalués. Cette période ne doit pas être nécessairement égale à la période de prévision. La commande "**Set options**" permet d'ajuster certains paramètres (fixés par défaut) spécifiques de la

procédure d'optimisation numérique utilisée pour l'estimation des paramètres des modèles, tels que le nombre maximal d'itérations, les critères de convergence, etc.

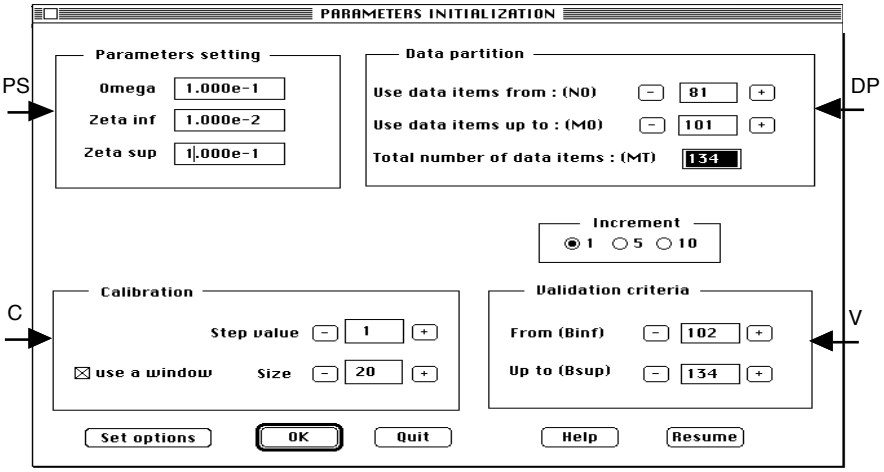


Figure 21. *Initialisation des paramètres nécessaires à l'application des modèles*

Les courbes correspondant aux résultats des évaluations sont affichées à la demande de l'utilisateur ; un exemple de courbes est donné à la figure 22. Les critères de validation des résultats des modèles sont fournis conjointement avec les résultats numériques.

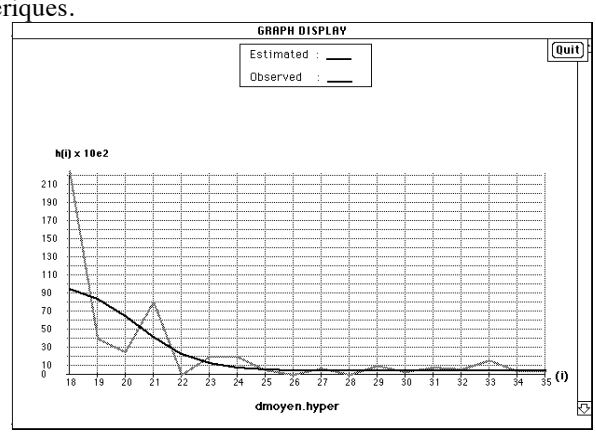


Figure 22. *Résultats graphiques*

8. Conclusion

Nous avons présenté une méthode et un outil permettant de mener des analyses et des évaluations de la fiabilité du logiciel basés sur des données collectées sur le système. Nous avons développé les différentes étapes de la méthode en mettant l'accent sur les objectifs et les résultats de chacune de ces étapes. Nous avons

également illustré certains aspects de notre approche sur des données collectées sur un système réel. Par ailleurs, nous avons appliqué la méthode proposée avec succès à plusieurs cas réels dont une synthèse est présentée sur la figure 23.

Système	Langage	Taille	Durée	Phases	# Syst.	# RD et/ou RC
E10-B	Assembleur	100 k-oct	3 ans	Val. / Op.	1400	58 RD / 136 RC
ESS1	Assembleur	300 k-oct	27 mois	Val. / Op.	15	465 RD/RC
ESS2	Assembleur	350 k-oct	32 mois	Val. / Op.	42	210 RD/RC
ESS3	Assembleur	420 k-oct	47 mois	Op.	37	212 RD/RC
ESS4	CHILL	815 k-Li	68 mois	Val. / Op.	146	3063 RD/RC
Equip. Tél.	PLM-86	5 10^5 inst.	16 mois	Val.	4	2150 RD
ST	variés	--	4 ans	Op.	1	414 RD

Equip. Tél. : équipement de Télécommunications — ST : Station de travail
Val. : validation — Op. : opération
RD : rapport de défaillance — RC: rapport de correction

Figure 23. *Caractéristiques de quelques logiciels étudiés*

Les logiciels ESS1, ESS2 et ESS3 correspondent à trois générations successives d'un même produit. Les études comparatives effectuées dans [KAA 93] pour les deux premiers, et dans [KAA 94] pour l'ensemble des trois, ont permis en plus d'analyser l'influence du processus de développement sur l'évolution de la fiabilité des générations[3].

[3] Ces études comparatives de plusieurs générations s'insèrent dans une approche plus générale : l'approche produit-processus introduite dans

Les applications présentées à la figure 23, conduites pendant une dizaine d'années et facilitées par l'utilisation de SoRel, nous ont permis d'améliorer notre méthode au fur et à mesure des applications. Au vu des résultats obtenus et forts de notre expérience cumulée, nous estimons que notre méthode a atteint un niveau de maturité suffisant qui permet de l'intégrer dans les programmes de contrôle de qualité du logiciel dans les milieux industriels.

[LAP

92] .

Bibliographie

[ADA 84] N. ADAMS, "Optimizing Preventive Service of Software Products", *IBM Journal of Research and Development*, 28 (1), pp. 2-14, 1984.

[ASC 84] H. ASCHER et H. FEINGOLD, *Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes,* Vol. 7, 1984.

[BAS 84] V. R. BASILI et D. M. WEISS, "A Methodology for Collecting Valid Software Engineering Data", *IEEE Transactions on Software Engineering*, 10 (6), pp. 728-738, 1984.

[COM 89] P. COMER, "Reliability Data Collection and the Software Data Library", dans Actes *6th EUREDATA Conf.,* pp. 824-839, Sienna, Italy, 1989.

[COX 66] D. R. COX et P. A. W. LEWIS, *The Statistical Analysis of Series of Events,* Monographs on Applied Probability and Statistics, Chapman and Hall, London, 1966.

[GAU 90] O. GAUDOIN, *Outils statistiques pour l'évaluation de la Fiabilité des logiciels,* Thèse de Doctorat, Université Joseph Fourier, Grenoble 1, 1990.

[GOE 79] A. L. GOEL et K. OKUMOTO, "Time-Dependent Error Detection Rate Model for Software and Other Performance Measures", *IEEE Trans. on Reliability*, R-28, pp. 206-211, 1979.

[GRA 87] R. B. GRADY et D. L. CASWELL, *Software Metrics: Establishing a Company-Wide Program,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, USA, 1987.

[IYE 82] R. IYER, S. E. BUTNER et E. J. Mcluskey, "A Statistical Failure/Load Relationship: Results of a Multicomputer Study", *IEEE Trans. on Computers*, C-31 (4), pp. 697-706, 1982.

[KAA 90] M. KAANICHE, K. KANOUN et S. METGE, "Analyse des défaillances et suivi de la validation du logiciel d'un équipement de télécommunication", *Annales des télécommunications*, 45 (11-12), pp. 657-670, 1990.

[KAA 93] M. KAANICHE et K. KANOUN, "Software Failure Data Analysis of two Successive Generations of a Switching System", dans Actes *12th Int. Conf. on Computer Safety, Reliability and Security (SAFECOMP'93),* pp. 230-239, Poznan-Kiekrz, Poland, 1993.

[KAA 94] K. KAANICHE, K. KANOUN, M. Cukier et M. Bastos Martini, "Software Reliability Analysis of Three Successive Generations of a Switching System", dans Actes *First European Conference on Dependable Computing (EDCC-1),* pp. 473-490, Berlin, Germany, 1994.

[KAN 87] K. KANOUN et T. SABOURIN, "Analyse des Défaillances et Evaluation de la Sûreté de Fonctionnement du Logiciel d'un Autocommutateur Téléphonique", *Technique et Science Informatique*, 6 (4), pp. 287-304, 1987.

[KAN 91] K. KANOUN, M. BASTOS MARTINI et J. MOREIRA de SOUZA, "A Method for Software Reliability Analysis and Prediction—Application to The TROPICO-R Switching System", *IEEE Trans. Software Engineering*, SE-17 (4), pp. 334-344, 1991.

[KAN 92] K. KANOUN, M. KAANICHE, J.-C. LAPRIE et S. METGE, "SoRel : outil d'analyse et d'évaluation de la fiabilité du logiciel à partir de données statistiques de défaillance", dans Actes *8ème Colloque de Fiabilité et de Maintenabilité,* pp. 422-431, Grenoble, France, 1992.

[KAN 93] K. KANOUN, M. KAANICHE et J.-C. LAPRIE, "Experience in Software Reliability: From Data Collection to Quantitative Evaluation", dans Actes *4th Int. Symp. on Software Reliability Engineering (ISSRE'93),* pp. 234-245, Denver, CO, USA, 1993.

[KAN 94] K. KANOUN et J.-C. LAPRIE, "Software Reliability Trend Analyses: from Theoretical To Practical Considerations", *IEEE Transactions on Software Engineering*, 20 (9), pp. 740-747, 1994.

[KAN 95] K. KANOUN et J.-C. LAPRIE, "Trend Analysis", dans *Software Reliability Engineering Handbook,* (M. Lyu, Ed.), Ch. 10, Mc Graw Hill, 1995.

[KEN 92] G. Q. KENNEY et M. A. VOUK, "Measuring the Field Quality of Wide-Distribution Commercial Software", dans Actes *3rd Int. Symp. on Software Reliability Engineering (ISSRE'92),* pp. 351-357, Raleigh, NC, USA, 1992.

[LAP 91] J.-C. LAPRIE, K. KANOUN, C. BEOUNES et M. KAANICHE, "The KAT (Knowledge-Action-Transformation) Approach to the Modeling and Evaluation of Reliability and Availability Growth", *IEEE Trans. Software Engineering*, SE-17 (4), pp. 370-382, 1991.

[LAP 92] J.-C. LAPRIE, "For a Product-in-a Process Approach to Software Reliability Evaluation", dans Actes *3rd Int. Symp. on Software Reliability Engineering,* pp. 134-139, Raleigh, NC, USA, 1992.

[LAR 94] A. LARYD, "Operating Experience of Software in Programmable Equipment Used in ABB Atom Nuclear I&C Applications", dans Actes *IAEA, Technical Committee Meeting,* pp. 1-12, Helsinki, Finland, 1994.

[LEV 90] Y. LEVENDEL, "Reliability Analysis of Large Software Systems: Defects Data Modeling", *IEEE Trans. Software Engineering*, SE-16 (2), pp. 141-152, 1990.

[MUS 87] J. D. MUSA, A. IANNINO et K. OKUMOTO, *Software Reliability: Measurement, Prediction, Application,* Computer Science Series, McGraw-Hill, New-York, 1987.

[OHB 84] M. OHBA, "Software Reliability Analysis Models", *IBM Journal of Research and Development*, 21 (4), pp. 428-443, 1984.

[ROS 89] N. ROSS, "The Collection and Use of Data for Monitoring Software Projects", dans *Measurement for Software Control and Assurance,* (B. A. Kitchenham and B. Littlewood, Ed.), pp. 125-154, Elsevier Applied Science, London and New York, 1989.

[TRO 86] R. TROY et Y. ROMAIN, "A Statistical Methodology for the Study of the Software Failure Process and Its Application to the ARGOS Center", *IEEE Trans. on Software Engineering*, SE-12 (9), pp. 968-978, 1986.

[XIE 91] M. XIE, *Software Reliability Modeling,* World-Scientific, Singapore, 1991.