COLLECTE ET ANALYSE DE DONNÉES RELATIVES A LA SÛRETÉ DE FONCTIONNEMENT DU LOGICIEL

Karama Kanoun

Journées nationales ISdF "Retour d'expérience et banques de données" 24-25 Mars 1992 Paris.

COLLECTE ET ANALYSE DE DONNÉES RELATIVES A LA SÛRETÉ DE FONCTIONNEMENT DU LOGICIEL

Karama KANOUN

LAAS-CNRS
7, avenue du Colonel Roche, 31077 Toulouse Cedex - FRANCE

RÉSUMÉ

L'objectif de cette communication est d'analyser les raisons de l'absence de banques de données de sûreté de fonctionnement pour le logiciel, de présenter brièvement les données à collecter en vue d'aboutir à une telle banque de données et de montrer quelques exemples de résultats obtenus à partir de l'analyse et du traitement de données collectées sur des logiciels réels.

INTRODUCTION

Il n'existe pas de banque de données relative au logiciel. Il y a eu certes un certain nombre de tentatives de collecte de données mises en place en grande partie à l'initiative de groupes de normalisation (nationaux ou internationaux) ou par des projets européens, mais une organisation en banque de données ouverte n'a pas encore été effectuée à ce jour et ce pour un certain nombre de raisons et de difficultés que nous essayerons d'analyser dans cette communication.

Les analyses des informations issues d'une collecte de données ont un double objectif : à court terme, et pour le logiciel concerné, permettre de réagir rapidement et efficacement sur son développement (guider le développement) et, à plus long terme, accumuler une connaissance approfondie des logiciels développés afin d'améliorer le développement et d'effectuer des prévisions de comportement pour les produits à venir (capitaliser l'expérience).

Cette communication se compose de quatre paragraphes. Le premier permet d'analyser les raisons de l'absence de banques de données pour le logiciel, le second présente brièvement les données à collecter en vue d'aboutir à une banque de données de sûreté de fonctionnement du logiciel et l'organisation d'une

telle collecte; les deux derniers paragraphes sont consacrés à l'analyse et au traitement de données de fiabilité de logiciels réels.

1- BANQUE DE DONNÉES DU LOGICIEL : PROBLÉMATIQUE, OBSTACLES

Une collecte de données (et de ce fait, une banque de données) doit être effectuée pour répondre à un ou plusieurs objectifs qui ont été clairement définis au départ. Pour le logiciel, les objectifs sont d'ordre multiple parmi lesquels on peut citer, sans être exhaustif :

- estimer le nombre de fautes[1*] que contient le logiciel afin de mieux appréhender l'effort de test qu'il est nécessaire de fournir pour aboutir à un logiciel "zéro faute" (souvent appelé "zéro défaut"),
- s'assurer que le logiciel est facilement "maintenable" à travers l'étude des mesures de complexité,
- gérer les configurations du logiciel aussi bien en développement qu'en vie opérationnelle,
- assurer le suivi du développement et mieux planifier en maîtrisant le passage d'une phase du développement à la suivante,
- évaluer la sûreté de fonctionnement du logiciel en vie opérationnelle afin d'effectuer des estimations globales de la sûreté de fonctionnement d'un système composé de matériel et de logiciel.

La nature et le type de données à collecter sont étroitement liés à l'objectif fixé et au traitement qui sera effectué, et donc au modèle sous-jacent à ce traitement.

Une collecte de données suppose en effet l'existence de modèle(s) et/ou de méthodes qui permettent le traitement de ces données afin d'aboutir aux objectifs de départ. Or, il n'existe pas à l'heure actuelle de modèles et de méthodes associés à tous les objectifs cités ci-dessus qui ont fait leur preuve. De plus, dû aux corrections qui sont effectuées sur le logiciel au fur et mesure des activations des fautes, le processus de défaillance du logiciel n'est pas répétitif, ce qui pose en plus des problèmes de validité des traitements statistiques de ces données.

A ceci s'ajoutent des difficultés inhérentes au logiciels lui-même et à son processus de développement. Parmi ces difficultés on peut citer :

- un logiciel peut être considéré comme un produit intellectuel et est de ce fait lié aux personnes qui le codent et le testent,
- un logiciel résulte de l'utilisation de méthodes et d'outils de codage et de test et est de ce fait étroitement lié au processus de production qui est lui-même fonction de la nature du logiciel (le logiciel d'une application critique nécessite l'utilisation de méthodes de développement plus rigoureuses qu'un logiciel de traitement de texte par exemple),
- les méthodes de développement varient d'une entreprise à une autre, ce qui pose le problème de la validité des résultats issus d'une entreprise pour d'autres entreprises.

[1] * Les définitions précises de faute, défaillance , etc sont données dans [Laprie 91b].

Ceci nous amène à penser qu'il est très difficile d'aboutir à une banque de données qui permettrait d'aboutir à une norme pour la fiabilité du logiciel du style de la norme MIL-HDBK-217 pour le matériel. Cependant, il n'est pas exclu d'aboutir à ce type de norme pour une entreprise donnée développant du logiciel pour des applications similaires et suivant le même processus de développement. La banque de données ainsi créée caractérisera le produit à travers la caractérisation du processus de développement.

Un autre problème majeur rencontré lors des tentatives d'établissement de banques de données pour le logiciel est la confidentialité des données. En effet un grand nombre d'entreprises disposent de données relatives aux logiciels qu'elles développent et/ou utilisent mais ces données ne peuvent être communiquées à l'extérieur afin d'éviter leur utilisation par des entreprises concurrentes.

Face à ces objectifs et à ces difficultés, il n'existe que peu de modèles et de méthodes permettant de caractériser le logiciel et son comportement et qui pourraient donner lieu à une banque de données du logiciel. Parmi ces méthodes et modèles on peut citer :

- des modèles de coût tels que celui présenté dans [Boehm 81],
- des mesures de complexité telles que le nombre cyclomatique de McCabe [McCabe 76], ou le nombre de branchements, mais le lien entre ces mesures et la fiabilité du logiciel d'une part et le lien entre ces mesures et la maintenabilité du logiciel d'autre part n'ont pu être établis de façon rigoureuse,
- des tests de tendance (croissance ou décroissance de fiabilité) basés sur des critères statistiques permettant de suivre l'évolution du logiciel et de guider le développement [Kanoun 86], [Kanoun 91b],
- des modèles de croissance de fiabilité qui ne donnent de bons résultats que s'ils sont appliqués dans des conditions particulières (entre autre en cas de croissance de fiabilité effective) [Kanoun 87].

Les tests de tendance et les modèles de croissance de fiabilité nécessitent une collecte de données du même type. En effet, l'application des modèles de croissance de fiabilité peut être basée sur les résultats des test de tendance, ce qui améliore considérablement les résultats de l'évaluation [Kanoun 91a]. La collecte de données fait l'objet du prochain paragraphe qui traitera de la définition de la collecte de données et des données à collecter, de l'organisation pratique et de quelques recommandations concernant sa mise en place.

2- COLLECTE DE DONNÉES DE FIABILITÉ, DÉFINITION, ORGANISATION

Données à collecter

Un compromis entre le nombre de données à collecter et le travail de collecte doit être étudié de près : disposer d'un grand nombre de données pourrait aider à mieux comprendre les phénomènes complexes mis en jeu mais alourdit de façon significative la tâche de collecte, et peut même décourager les personnes correspondantes. Les données à collecter sont de deux types :

- des données caractérisant le produit lui-même ainsi que le processus de développement et l'environnement d'utilisation : taille, langage, fonctions, version, méthodes et outils de développement, charge, … ,
- des données concernant les défaillances et les corrections introduites : dates ou instants auxquels elles ont eu lieu, nature, conséquence, composant logiciel concerné par la ou les corrections effectuées, … .

La collecte de données se fait généralement à l'aide de fiches avec des rubriques bien définies à remplir soit par la personne qui a constaté l'anomalie soit par la personne qui effectue les modifications. Ces fiches sont soit sur support informatique soit sur papier. Une collecte automatique est conseillée mais elle est loin d'être pratique courante à l'heure actuelle.

Un bon compromis est généralement atteint avec trois types de fiches de collecte : une fiche de défaillance, une fiche de correction et un journal de bord, chacun d'eux contenant des informations concernant le produit et son utilisation et des informations spécifiques concernant respectivement les défaillances, les corrections et les conditions d'utilisation. Le journal de bord recense les différents tests effectués ainsi que leurs résultats; il concerne de ce fait plus particulièrement le test. Les deux autres fiches sont à remplir aussi bien en développement qu'en vie opérationnelle. Ces fiches ne peuvent pas être définies dans l'absolu, elles sont à adapter pour chaque entreprise, en fonction du logiciel développé. Cependant, elles doivent contenir, selon l'objectif visé, les rubriques relatives à cet objectif. En fonction du nombre d'informations à recueillir, les fiches de défaillance et de correction peuvent être regroupées en une seule fiche avec des parties distinctes qui peuvent être remplies par des personnes différentes (lors de l'apparition de l'anomalie et lors de la correction par le service concerné).

A titre indicatif, le type d'information à recueillir en fonction des objectifs est le suivant :

- si l'objectif est d'étudier la fiabilité du logiciel par rapport à certains modes de défaillance ou par rapport à certaines tâches critiques, il est important de noter les conséquences des défaillances,
- si l'objectif est d'étudier la fiabilité du logiciel par rapport à ses composants ou même uniquement la fiabilité de quelques composants que l'on pense pouvoir ré-utiliser pour d'autres applications, il est important de noter la localisation des fautes par composant,
- si l'objectif est d'étudier l'influence de la charge sur le comportement, il faut noter les conditions qui ont entraîné la défaillance, ...

Il est à noter que si, lors du test, une fiche de défaillance par défaillance ainsi qu'une fiche de correction à chaque modification, constituent une grande charge — due au nombre relativement élevé de défaillances et de corrections — une collecte de données sous forme de "nombre de défaillances par unité de temps" peut être suffisante dans un premier temps. L'unité de temps peut varier d'une phase à l'autre (le jour pour les premières phases, la semaine ensuite par exemple). Une telle collecte ne permettra qu'une analyse de fiabilité très succincte mais qui peut être suffisante lors des premières phases du test.

Organisation et retombées économiques

La collecte de données doit être bien définie avant même qu'elle ne soit démarrée. Une définition claire des données à collecter est une condition nécessaire à sa réussite. La collecte de données doit être intégrée au processus de développement et l'attribution des rôles des différentes personnes chargées de la collecte doit être effectuée explicitement. Si une autre collecte de données est en cours pour d'autres objectifs (pour la gestion de configuration ou pour demande de correction à d'autres personnes impliquées dans le développement par exemple), bien définir les liens éventuels entre elles et les regrouper dans la mesure du possible.

Un élément primordial pour la réussite d'une collecte de données est la sensibilisation et "l'éducation" des personnes chargées de la collecte afin qu'elles puissent avoir une vue globale de la tâche à laquelle elles contribuent. En fait l'éducation doit être effectuée et prise en charge à tous les niveaux hiérarchiques car la préparation psychologique constitue un élément fondamental à l'acceptation de cette collecte. Un retour d'expérience assez rapide contribue efficacement à cette sensibilisation. Il faut surtout éviter de donner l'impression que l'objectif de la collecte de données est de tester la qualité du programmeur ou de contrôler son efficacité : on contrôle le produit et/ou le processus de production et non le programmeur ou le testeur.

La mise en place d'une collecte de données de fiabilité engendre — au premier abord — des coûts supplémentaires notamment durant le test et pourrait éventuellement constituer un obstacle à cette mise en place. Cependant le gain entraîné par l'analyse "à temps" des données observées est incommensurable comparé à ce sur-coût. En effet, il a été constaté expérimentalement que le coût d'une faute détectée en vie opérationnelle est au moins 100 fois supérieur au coût de cette même faute si elle avait été détectée dès les premières phases du développement [Boehm 81]. Le gain global résultant d'une collecte de données suivie d'une analyse est incontestable.

3- SÉLECTION ET TRAITEMENT DES DONNÉES COLLECTÉES

Lors de la collecte de données, il est souvent difficile d'identifier en temps réel la cause de la défaillance, matériel, logiciel, opérateur,… Un tri est généralement nécessaire avant toute étude de fiabilité afin de ne garder que les défaillances du logiciel [Kaâniche 90], [Levendel 90].

Dans le cas qui nous intéresse, une fois le tri effectué, le traitement consiste à appliquer :

- des tests de tendance pour suivre l'évolution du logiciel et mettre en évidence les dérives,
- éventuellement des modèles de fiabilité afin de vérifier que le niveau de fiabilité atteint par le logiciel est conforme à celui attendu ou de suivre l'évolution de la fiabilité dans le temps.

Objectifs des tests de tendance

Les tests de tendance sont de nature statistique et permettent d'apprécier l'efficacité des activités de test du logiciel et de contrôler leur progression [Kanoun 91b]. Ces tests constituent des **indicateurs** de fiabilité. Leur rôle est d'attirer l'attention sur des problèmes ou anomalies qui, sans l'utilisation de ces tests, pourraient se manifester mais beaucoup plus tard voire trop tard, ce qui permet de trouver des solutions à ces problèmes assez rapidement. Ils ne donnent pas directement la solution au problème.

Un exemple d'utilisation des résultats de tests de tendance est le suivant : une **décroissance de fiabilité** au début d'une nouvelle activité de test est généralement attendue et est considérée comme normale. Par contre, une décroissance de fiabilité durant une période de temps assez longue peut être alarmante et attirer l'attention sur une anomalie du développement. La recherche de la cause de cette anomalie pourrait éventuellement révéler des lacunes dans le processus de développement et aider à ré-orienter le cas échéant l'activité de test. Une telle analyse peut aussi contribuer à la prise de décision de ré-écrire 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** suivant 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 révéler un ralentissement de l'activité de test.

Une **fiabilité stabilisée** 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 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.

Enfin, les tests de tendance sont une aide indispensable pour l'application des modèles de croissance de fiabilité : une fois les zones de croissance et de décroissance de fiabilité identifiées, les modèles de croissance de fiabilité pourront être appliqués sur ces zones afin d'évaluer les mesures de sûreté de fonctionnement, si tel est l'objectif [Kanoun 89].

Quelques tests de tendance

Deux catégories de tests de tendance peuvent être distinguées : des tests graphiques et des tests analytiques [Ascher 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.

Les tests graphiques permettent de visualiser par exemple :

- le nombre cumulé de défaillances sur la période d'observation,
- le nombre de défaillances sur des intervalles de temps (ces intervalles sont appelés unités de temps),
- la moyenne des intervalles de temps entre défaillances : test de la moyenne arithmétique.

Ces tests graphiques sont souvent utilisés pour guider le développement [Ross 89] mais de façon intuitive. Les tests analytiques permettent de mieux quantifier la variation de fiabilité. Parmi les tests analytiques existant, le test de Laplace est le plus adapté. Ce test consiste à évaluer un indicateur statistique appelé coefficient de Laplace (u(k), où k représente l'unité de temps). En fonction de la valeur de u(k), on en déduit qu'il y a croissance ou décroissance de fiabilité avec un niveau de confiance donné. Sur le plan pratique, le signe de cet indicateur et son sens de variation donnent respectivement la tendance globale et la tendance locale comme indiqué sur la figure 1.

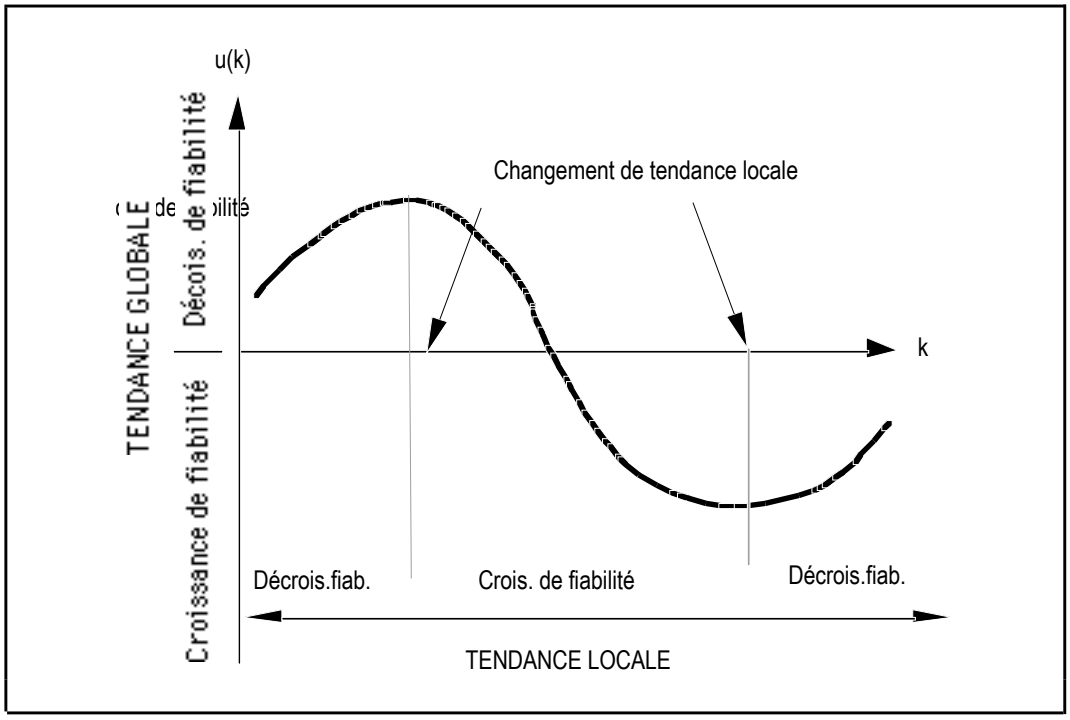

Figure 1: Coefficient de Laplace et variation de la tendance locale et globale

Modèles de fiabilité

Il existe un très grand nombre de modèles de croissance de fiabilité. Ces modèles permettent d'estimer soit de façon replicative soit de façon prévisionnelle les mesures de sûreté de fonctionnement. L'application d'un modèle consiste à estimer dans un premier temps les paramètres du modèle à l'aide des données collectées et d'une procédure d'inférence. Les paramètres du modèle sont ensuite remplacés par leurs valeurs numériques dans le modèle de fiabilité afin d'estimer les mesures de sûreté de fonctionnement [Kanoun 89]. L'utilisation de critères

statistiques de validation permet d'apprécier l'adéquation du modèle appliqué ; ces critères permettent aussi de comparer les résultats issus de différents modèles.

Mise en œuvre pratique

Les tests de tendance et les modèles de croissance de fiabilité peuvent être mis en œuvre à l'aide de logiciels tels que le logiciel SoRel [Kanoun 91c]. Ce logiciel permet d'une part d'effectuer le test de la moyenne arithmétique et de Laplace et d'autre part d'appliquer quatre modèles de croissance de fiabilité basés sur des hypothèses différentes de variation de la fiabilité. Les modèles mis en œuvre sont : le modèle exponentiel de Goel-Okumoto, le modèle S-Shaped de Yamada et al., le modèle de Littlewood-Verrall et le modèle hyperexponentiel de Kanoun-Laprie développé au LAAS.

4- EXEMPLES DE RÉSULTATS ISSUS D'ANALYSES DE FIABILITÉ BASÉES SUR UNE COLLECTE DE DONNÉES

Une méthode d'analyse et d'évaluation de la fiabilité du logiciel basée sur une collecte de données est définie dans [Kanoun 88b]. Elle est constituée d'une étape d'analyse pour assurer le suivi du logiciel et d'une étape d'évaluation. Cette méthode a été appliquée à des données collectées sur des logiciels en validation et/ou en vie opérationnelle. Les caractéristiques de ces logiciels sont résumées par la figure 2, les trois premiers systèmes sont des autocommutateurs téléphoniques.

Système	Langages	Volume	Durée de l'observation	Phases	Nombre de systèmes observés	Nombre de fiches de défaillance
E10-B	Assembleur	100 ko	3 ans	Val. / Op.	1400	58 + 136 fiches de correction
TROPICO-R 1500	Assembleur	300 ko	27 mois	Val. / Op.	15	461
TROPICO-R 4096	Assembleur	350 ko	32 mois	Val. / Op.	42	227
Equipement Télécom.	PLM-86	5 10\s\ up5(5) inst.	16 mois	Val.	4	2150
Station de travail	variés	--	4 ans	Op.	1	227

Val. : validation Op. : vie opérationnelle

Figure 2: Caractéristiques des logiciels pour lesquels une étude de fiabilité basée sur une collecte de données a été effectuée.

A titre indicatif, les résultats obtenus pour les trois autocommutateurs sont :

- pour le E10-B, l'évaluation basée sur la collecte de données a permis d'estimer le taux de défaillance du logiciel en vie opérationnelle afin

d'effectuer une évaluation de la sûreté de fonctionnement du système en tenant compte des défaillances matérielles et logicielles,

- pour les TROPICO-R, deux approches complémentaires selon deux points de vue différents ont été adoptées :
  - du point de vue du concepteur, estimation de l'effort de maintenance à assurer durant la vie opérationnelle afin de satisfaire les demandes de correction en provenance des différents sites,
  - du point de vue de l'utilisateur, estimation du taux de défaillance moyen pour un utilisateur durant la vie opérationnelle.

CONCLUSION

La collecte et l'analyse de données de fiabilité du logiciel ne constitue qu'une étape permettant d'aboutir à une banque de données par entreprise. En effet il reste encore du travail aussi bien sur le plan théorique que pratique pour aboutir à la caractérisation du produit à travers la caractérisation de son processus de développement.

La ré-utilisation de composants à l'intérieur d'une même entreprise prend d'ampleur. La première étape pourrait consister à établir une banque de données pour les composants. L'estimation de la sûreté de fonctionnement du logiciel en fonction de celle de ses composants est alors effectuée selon une approche de modélisation multi-composant comme celle présentée dans [Laprie 91a].

Enfin, signalons que l'objectif final de toute analyse et traitement de données de fiabilité est d'aboutir à une prévision du comportement d'un système en tenant compte à la fois du matériel et du logiciel. Ceci pourrait être effectué grâce à une approche de modélisation commune prenant en compte à la fois les aspects du matériel et du logiciel [Laprie 92].

REMERCIEMENTS

Les réflexions menées dans cette communication ont bénéficié des travaux effectués avec Jean-Claude Laprie du LAAS ; je tiens à le remercier. Je tiens également à remercier Christian Béounes et Alain Costes du LAAS pour leurs critiques constructives.

RÉFÉRENCES

Ascher 84 H.Ascher, H.Feingold, *Repairable systems reliability: modeling, inference, misconceptions and their causes*, Lecture notes in statistics, Vol. 7, 1984.

Boehm 91 B.Boehm, *Software engineering economics*, Prentice Hall, Englewood Cliffs, New York, 1981.

Kaâniche 90M.Kaâniche, K.Kanoun, S.Metge, "Analyse des défaillances et suivi de la validation du logiciel d'un équipement de télécommunications", *Annales des Télécommunications*, Vol. 45, N.11-12, pp. 657-670, Nov.-Déc. 1990.

Kanoun 87 K.Kanoun, T.Sabourin, "Analyse des défaillances et évaluation de la fiabilité du logiciel d'un autocommutateur téléphonique", *Technique et Science Informatique* (TSI), Vol. 6, N.4, 1987, pp. 287-303.

Kanoun 88a K.Kanoun, "Analyse de la croissance de fiabilité", Acte de 6ième *Colloque de Fiabilité et de Maintenabilité*, Strasbourg, Oct. 1988, pp. 651-656.

Kanoun 88b K.Kanoun, J.C.Laprie, T.Sabourin, "A method for software reliability growth analysis and assessment", Acte de *Software Engineering & its Applications*, Toulouse, France, Déc. 1988, pp. 859-878.

Kanoun 89 K.Kanoun, "Croissance de la sûreté de fonctionnement des logiciels : caractérisation, modélisation, évaluation", Thèse de Docteur ès-Science, Institut National Polytechnique de Toulouse, Sept. 1989.

Kanoun 91a K.Kanoun, M.Bastos Martini, J.Moreira De Souza, "A method for software reliability analysis and prediction — Application to the TROPICO-R switching System", *IEEE Transactions on Software Engineering,* Avril 1991, pp. 334-344.

Kanoun 91b K.Kanoun, J.C.Laprie, "The role of trend analysis in software development and validation", *IFAC Int. Conf. on Safety, Security and Reliability (SAFECOMP'91)*, Trondheim, Norway, 30 Oct.-1 Nov. 1991, pp. 169-172.

Kanoun 91c K.Kanoun, M.Kaâniche, J.C.Laprie, S.Metge, "Méthodes et modèles mis en œuvre dans le logiciel SoRel — Software Reliability", rapport de recherche LAAS N. 91-211.
SoRel est commercialisé par CEP-Systèmes Toulouse.

Laprie 91a J.C.Laprie, K.Kanoun, C.Béounes, M.Kaâniche, "The KAT — Knowledge-Action-Transformation) — Approach to the modeling and evaluation of reliability and availability growth", *IEEE Transactions on Software Engineering*, Vol.17, N.4, Avril 1991, pp. 370-382.

Laprie 91b J.C.Laprie, "*Dependability: basic concepts and terminology*", *Dependable Computing and Faul-Tolerant Systems,* Vol. 5, Editeur J.C.Laprie, Springer-Verlag 1991.

Laprie 92 J.C.Laprie, K.Kanoun, "X-Ware Reliability and Availability Modeling", *IEEE Transactions on Software Engineering*, Vol. 2, Février 1992.

McCabe 76 T.J.McCabe, "'A complexity measure", *IEEE Transactions on Software Engineering*, Vol. 2, N. 4, Déc. 1976, pp. 308-320.

Levendel 91 Y.Levendel, "Software quality improvement process: when to stop testing", Acte de *Software Engineering & its Applications*, Toulouse, France, Déc. 1991, pp. 729-749.

Ross 89 N.Ross, "The collection and use of data for monitoring software projects", Measurement for software control and assurance, Edité par B.A.Kitchenham et B.Littlewood, Elsevier Applied Science, Londres et New York, pp. 125-154.

Valette 88 V.Valette, "An environment for software reliability evaluation", Acte de *Software Engineering & its Applications*, Toulouse, France, Déc. 1988, pp. 879-897.