FAUTES ACCIDENTELLES ET FAUTES INTENTIONNELLES : UNE PERSPECTIVE

Jean-Claude Laprie LAAS-CNRS

Yves Deswarte LAAS-CNRS et INRIA

7, Avenue du Colonel Roche — 31077 Toulouse

Résumé *- Cette communication a pour objet d'examiner les différences et similitudes existant entre deux grandes sources de défaillance des systèmes informatiques, les fautes accidentelles, commises ou survenant sans malveillance, et les fautes intentionnelles, commises délibérément dans l'intention de nuire. Ces sources de défaillance constituent deux classes d'entraves à la sûreté de fonctionnement, un concept générique englobant et généralisant les notions classiques de fiabilité, disponibilité, sécurité vis-à-vis des défaillances catastrophiques, et sécurité vis-à-vis des accès ou modifications non-autorisées de l'information.*

Introduction

La sûreté de fonctionnement telle que définie dans [Lap 92] est un concept générique englobant, outre fiabilité et disponibilité, les notions de sécurité vis-à-vis des défaillances catastrophiques, c'est-à-dire la sécurité-innocuité, et vis-à-vis des accès ou modifications non-autorisées de l'information, c'est-à-dire la sécurité-confidentialité, toutes notions qui sont vues comme ses attributs. La sûreté de fonctionnement fournit donc un cadre conceptuel pour exprimer les relations souvent antagonistes existant entre fiabilité et disponibilité d'une part, sécurité-innocuité ou sécurité-confidentialité d'autre part. Ces relations découlent des similitudes et différences existant entre les causes de défaillance habituellement considérées pour fiabilité, disponibilité et sécurité-innocuité, que nous qualifions de *fautes accidentelles*, et celles considérées pour la sécurité-confidentialité, que nous qualifions de *fautes intentionnelles*.

Cette communication comprend quatre parties. Après avoir dans une première partie rappelé les définitions de base relatives à la sûreté de fonctionnement, nous traitons dans la deuxième partie des entraves à la sûreté de fonctionnement, c'est-à-dire les fautes, erreurs et défaillances, en focalisant l'attention sur la distinction entre fautes accidentelles et fautes intentionnelles. La troisième partie est consacrée aux relations entre les attributs de la sûreté de fonctionnement, ce qui nous permet de revenir dans la quatrième partie sur la relation entre défaillance et spécification.

Complexité et répartition des systèmes informatiques permettent de fournir des services de plus en plus riches aux utilisateurs, mais en même temps accroissent notre vulnérabilité aux défaillances de ces systèmes, que ces défaillances soient dues à des fautes accidentelles ou intentionnelles. Il y a

donc un besoin urgent de dépasser les clivages traditionnels et de porter une vue globale sur les défaillances des systèmes informatiques, ce à quoi s'attache cette communication.

1- Le concept de Sûreté de Fonctionnement

Les principales définitions relatives aux entraves, moyens et attributs de la sûreté de fonctionnement [Lap 92] sont résumées sur la figure 1, ainsi que sous une forme arborescente sur la figure 2.

La **sûreté de fonctionnement** d'un système informatique est la propriété qui permet à ses utilisateurs de *placer une confiance justifiée dans le service* qu'il leur délivre. Le **service** délivré par un système est son comportement *tel que perçu* par son, ou ses, utilisateurs ; un **utilisateur** est un autre système (humain ou physique) qui *interagit* avec le système considéré.

Selon la, ou les applications auxquelles le système est destiné, l'accent peut être mis sur différentes facettes de la sûreté de fonctionnement, ce qui revient à dire que la sûreté de fonctionnement peut être vue selon des propriétés différentes mais complémentaires, qui permettent de définir ses *attributs* :

- par rapport au fait d'être *prêt à l'utilisation*, la sûreté de fonctionnement est perçue comme la **disponibilité**,
- par rapport à la *continuité du service*, la sûreté de fonctionnement est perçue comme la **fiabilité**,
- par rapport à la *non-occurrence de défaillances catastrophiques*, la sûreté de fonctionnement est perçue comme la **sécurité**-innocuité,
- par rapport à la *prévention d'accès ou de manipulations non-autorisées de l'information*, la sûreté de fonctionnement est perçue comme la **sécurité**-confidentialité.

Une **défaillance** du système survient lorsque le service délivré n'est plus conforme à la spécification, la **spécification** étant une description agréée de la fonction ou du service attendu du système. Une **erreur** est la partie de l'état du système qui est susceptible d'entraîner une défaillance : une erreur affectant le service est une indication qu'une défaillance survient ou est survenue. La cause adjugée ou supposée d'une erreur est une **faute**.

Le développement d'un système sûr de fonctionnement passe par l'utilisation *combinée* d'un ensemble de méthodes qui peuvent être classées en :

- **prévention des fautes** : comment empêcher l'occurrence ou l'introduction de fautes,
- **tolérance aux fautes** : comment fournir un service conforme à la spécification en dépit des fautes,
- **élimination des fautes** : comment réduire la présence (nombre, sévérité) des fautes,
- **prévision des fautes** : comment estimer la présence, la création et les conséquences des fautes.

Prévention des fautes et tolérance aux fautes peuvent être vues comme constituant l'**obtention** de la sûreté de fonctionnement : comment *fournir* au système l'aptitude à délivrer un service conforme à la spécification ; élimination des fautes et prévision des fautes peuvent être vues comme constituant la **validation** de la sûreté de fonctionnement : comment *avoir confiance* dans l'aptitude du système à délivrer un service conforme à la spécification.

Les notions qui ont été introduites peuvent être groupées en trois classes :

- les **entraves** à la sûreté de fonctionnement : fautes, erreurs, défaillances; elles sont les circonstances indésirables — mais non inattendues — causes ou résultats de la non-sûreté de fonctionnement (dont la définition se déduit simplement de celle de la sûreté de fonctionnement : la confiance ne peut plus, ou ne pourra plus, être placée dans le service délivré) ;
- les **moyens** pour la sûreté de fonctionnement : prévention des fautes, tolérance aux fautes, élimination des fautes, prévision des fautes ; il s'agit des méthodes et techniques permettant de fournir au système l'aptitude à délivrer un service conforme à la spécification, et de donner confiance dans cette aptitude ;
- les **attributs** de la sûreté de fonctionnement : disponibilité, fiabilité, sécurité-innocuité, sécurité-confidentialité ; ils permettent a) d'exprimer les propriétés qui sont attendues du système, et b) d'apprécier la qualité du service délivré, telle que résultant des entraves et des moyens de s'y opposer.

Figure 1 - Définitions de base relatives à la sûreté de fonctionnement

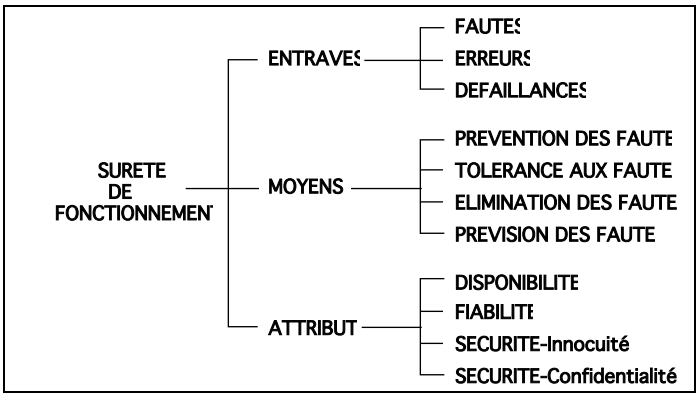

Figure 2 - L'arbre de la sûreté de fonctionnement

2- Les entraves à la sûreté de fonctionnement : fautes, erreurs, défaillances

2.1- Classes de fautes et de défaillances

Les entraves à la sûreté de fonctionnement sont de première importance, puisque nous devons bien évidemment commencer par cerner les circonstances indésirables auxquelles nous sommes, ou serons, confrontés. La figure 2 résume la classification des fautes : la part supérieure de la figure indique les points de vue permettant de définir de façon arborescente les classes élémentaires de fautes, et la partie inférieure donne les combinaisons vraisemblables selon ces points de vue, ainsi que les étiquettes usuelles associées à ces combinaisons.

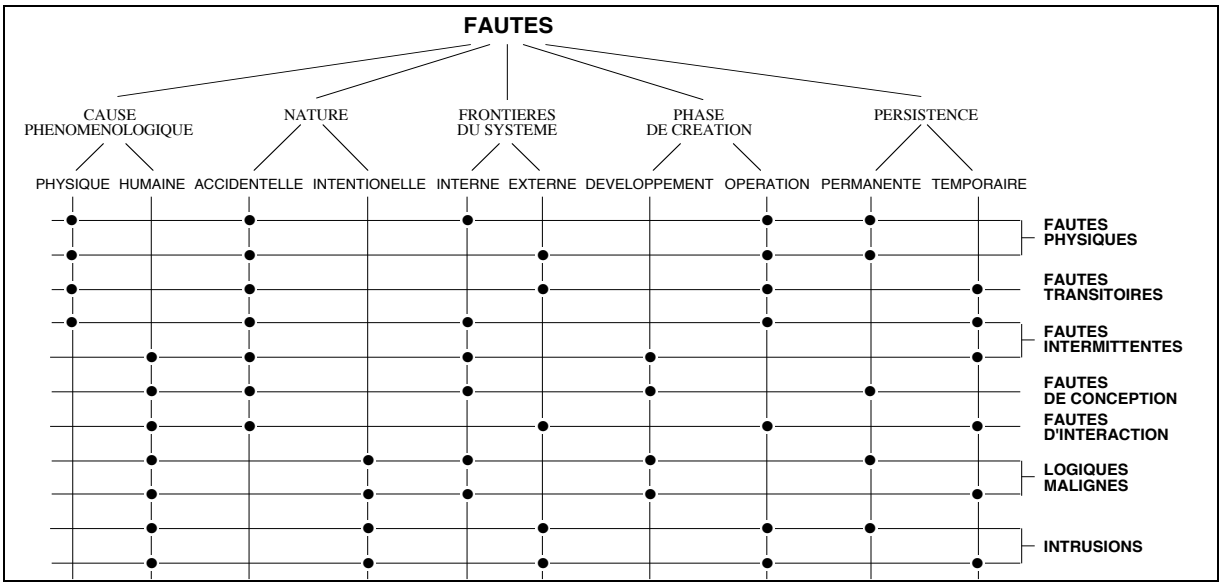

Figure 3 - Classes de fautes

Il est à noter que la notion même de faute est arbitraire, en ce sens qu'elle n'est autre qu'une facilité fournie pour permettre l'arrêt de la récursion induite par la relation causale entre fautes, erreurs et défaillances. D'où la définition donnée : cause *adjugée ou supposée* d'une erreur. Cette cause peut varier en fonction du point de vue adopté : mécanismes de tolérance aux fautes, équipes de maintenance, concepteurs, physiciens du solide, etc. En fait, *une faute n'est autre que la conséquence de la défaillance d'un système qui a délivré (y compris les concepteurs) ou qui délivre un service au système considéré*. Un système informatique étant une création humaine, toute faute l'affectant est de façon ultime due aux humains en ce sens qu'elle est la conséquence de notre

inaptitude à maîtriser tous les phénomènes qui régissent le comportement d'un système. En poursuivant le raisonnement, *toute faute peut être considérée comme une faute de conception permanente*. Ceci est effectivement vrai de façon ultime, mais n'est probablement pas très utile pour les développeurs et évaluateurs de systèmes, d'où l'utilité en l'état actuel de nos connaissances des diverses classes de fautes qui sont considérées dans la figure 3.

Un système ne défaille généralement pas toujours de la même façon, d'où la notion de modes de défaillance, qui peuvent être caractérisés selon trois points de vue comme indiqué sur la figure 4.

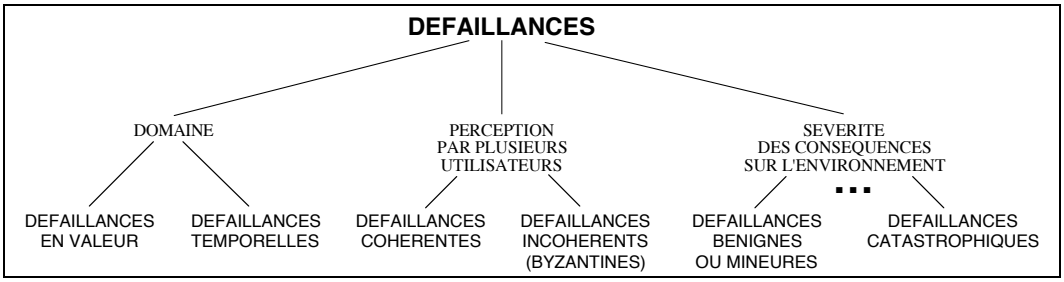

Figure 4 - Classes de défaillances

Une classe de défaillances relative à la fois aux valeurs et aux conditions temporelles est constituée par les *défaillances par arrêt* : l'activité du système, si tant est qu'il en ait une, n'est plus perceptible aux utilisateurs. Selon la forme que revêtent les interactions entre le système et ses utilisateurs, cette absence d'activité peut se traduire par un *figement* de l'*état* des sorties (valeur constante du service délivrée, cette valeur constante pouvant être la dernière valeur correcte, une valeur prédéterminée, …), ou par l'absence d'*événement* (par exemple pas d'envoi de message dans un système réparti), et donc par un *silence*. Un système dont toutes les défaillances sont — ou plus généralement dont les défaillances sont dans une mesure acceptable — des défaillances par arrêt est un *système à arrêt sur défaillance* ; dans les cas respectifs de figement et de silence, on est conduit à des *systèmes figés sur défaillance* ou à des *systèmes à silence sur défaillance* [Pow 88]. Selon les applications, une ou plusieurs classes de défaillance de sévérité intermédiaire entre défaillances bénignes et défaillances catastrophiques sont considérées.

2.2- Fautes accidentelles et fautes intentionnelles

Les fautes humaines ont une part prépondérante dans les causes de défaillance. Ainsi, ces dernières années, de nombreuses catastrophes ont été attribuées à des fautes humaines : Three Mile Island, Tchernobyl, Bhopal, l'Exxon Valdez, mais aussi la plupart des catastrophes aériennes. Parmi les fautes humaines, il est courant de distinguer les fautes accidentelles des fautes intentionnelles : les premières sont créées par méprise, alors que les secondes sont le résultat d'une volonté délibérée.

Les fautes humaines accidentelles sont directement liées aux *facteurs humains*. Elles peuvent être sommairement classées en :

- manques d'attention ou négligences, c'est-à-dire le résultat d'un défaut d'attention du créateur de la faute ; cela peut aller d'une simple faute de frappe ou de la non-prise en compte d'une alarme jusqu'à une suite d'actions inappropriées qu'aucun concepteur n'aurait pu imaginer ; une meilleure ergonomie des systèmes de développement (pour les fautes de conception) ou des interfaces homme-machine (pour les fautes d'interaction) peut permettre de réduire le taux d'occurrence de ces fautes, mais ne permet pas de les éviter totalement, car les phénomènes psychologiques mis en jeu sont très complexes ;

- manques de connaissance ou de compétence, correspondant à l'inadéquation du personnel à son poste de travail ; des actions de formation peuvent bien sûr remédier à ce type de problèmes ;
- d'autres fautes liées aux interactions homme-machine, pour lesquelles l'opérateur soit n'a pas à sa disposition suffisamment d'information sur le processus qu'il contrôle, soit n'a pas suffisamment de moyens d'action, soit encore se représente mal, mentalement, le processus en cours, ce qui le conduit à commettre des actions erronées ou à omettre d'agir à bon escient ; il s'agit dans ce cas autant de fautes de conception du système que de fautes d'interaction, puisqu'une bonne ergonomie a pour but d'éviter de telles fautes.

Ce classement sommaire est bien sûr un peu arbitraire et dans bien des cas, il sera difficile d'identifier à quelle classe appartient quelle faute. Par ailleurs, il est souvent délicat de faire la part des fautes de conception et des fautes d'opération. Par exemple, si un opérateur, attentif et compétent, suit scrupuleusement une procédure erronée pour traiter un incident prévu, c'est bien sûr une faute de conception de la procédure. Pourtant, bien souvent le rôle de l'opérateur sera aussi de faire face à des incidents imprévus, voire d'interrompre ou de modifier les procédures s'il constate que leur effet sur le système n'est pas celui qu'il attend. Il peut ainsi compenser, par une action bénéfique, une faute de conception (procédures incorrectes ou incomplètes). Mais s'il n'effectue pas cette action salutaire, doit-on pour autant lui attribuer une faute d'opération ?

Les fautes intentionnelles, quant à elles, peuvent être très distinctement classées selon la motivation de leurs créateurs :

- Si la faute a été créée avec la volonté de nuire ou de profiter du système indûment, il s'agit bien sûr d'une faute maligne ; l'objectif peut être :
  - d'obtenir, sans y être autorisé, des informations confidentielles ou de divulguer des informations confidentielles à des personnes non autorisées ; il s'agit alors d'attaques contre la confidentialité ;
  - de modifier les informations et/ou le service fourni par le système, en vue soit d'en tirer avantage (fraude), soit de nuire aux utilisateurs (déni de service) ; il s'agit alors d'attaques contre l'intégrité et la disponibilité ;
  - d'utiliser le système à des fins personnelles, sans y être autorisé ; si cette utilisation perturbe le fonctionnement vis-à-vis des utilisateurs autorisés, il s'agit d'une attaque contre la disponibilité ; en revanche, si cette utilisation n'a pas de conséquence sur le fonctionnement normal, il y a atteinte contre l'*exclusivité*, c'est-à-dire la propriété selon laquelle seuls les utilisateurs autorisés peuvent agir sur le fonctionnement du système.

  Dans ces trois cas, la faute peut-être une faute de conception, souvent appelée *logique maligne* (cheval de Troie, bombe logique, porte dérobée, etc.) ou une faute d'interaction, généralement appelée intrusion[1]. Par intrusions, il faut entendre non seulement les pénétrations par des individus non autorisés (*intrus externes*), mais aussi l'usage illégitime du système par des utilisateurs autorisés (*intrus internes*). Dans ce dernier cas, il s'agit pour l'intrus interne soit d'excéder ses privilèges (en exécutant des actions qui ne lui sont pas autorisées), soit d'abuser de ses privilèges, c'est-à-dire de commettre des actions illégitimes quoiqu'autorisées[2]. Ainsi, un opérateur, autorisé à arrêter un système

---

[1] Les virus, les vers et certains chevaux de Troie sont le résultat de la combinaison d'une faute de conception (réalisation du code du virus, du ver ou du cheval de Troie) et d'une intrusion (insertion du virus, ver ou cheval de Troie dans le système)

[2] Notons que du point de vue technique, un intrus externe devra contourner ou tromper les mécanismes d'authentification et d'autorisation, alors qu'un intrus interne excédant ses privilèges ne devra que contourner les mécanismes de protection. Un intrus interne abusant de ses privilèges n'a bien sûr pas à contourner les mécanismes

informatique, peut abuser de ce privilège pour empêcher (de façon illégitime) le fonctionnement normal du système. De même un administrateur peut être autorisé à modifier des mots de passe ou des droits d'accès et abuser de ce privilège pour obtenir des informations confidentielles.

- Au contraire, une "faute" de conception ou d'opération peut être le résultat d'un choix délibéré, pour obtenir un moindre mal ou par suite d'un compromis d'ingénierie. Par exemple, le concepteur d'un système embarqué dans un satellite peut ne pas ignorer que son système sera soumis à des flux de particules de haute énergie susceptibles de provoquer un mauvais fonctionnement du système ; il pourrait l'en protéger par un blindage, mais dans ce cas il ne serait pas possible de lancer le satellite ; il pourra donc être amené à délibérément supprimer ce blindage et accepter le risque induit par ces particules (ou choisir de mettre en œuvre des mécanismes de tolérance aux fautes adaptés). Des choix analogues doivent être faits au sujet des politiques d'autorisation : une politique trop stricte peut être inacceptable par les contraintes imposées aux utilisateurs, alors qu'une politique trop laxiste facilite les intrusions. Ce type de choix ne porte pas seulement sur les phases de conception, mais aussi sur la vie opérationnelle du système ; ainsi, un opérateur peut décider de commander (abusivement) un arrêt d'urgence plutôt que de prendre le risque d'un fonctionnement hors limite.

Mais cette distinction entre fautes intentionnelles bienveillantes et malveillantes peut être considérée comme bien arbitraire, puisque seule l'intention de leur auteur permet de les distinguer. Par exemple, une porte dérobée peut être insérée dans le but de permettre des intrusions ou dans celui de faciliter la maintenance. De même, il est souvent difficile de décider si une faute donnée est accidentelle ou intentionnelle. Ainsi, le choix délibéré d'un moindre mal ou d'un compromis d'ingénierie peut être jugé par la suite regrettable et donc considéré comme une faute accidentelle. Par ailleurs, l'auteur d'une faute intentionnelle maligne cherchera souvent à cacher ses mauvaises intentions en simulant une faute accidentelle.

Malgré ce côté arbitraire, la distinction entre fautes accidentelles et fautes intentionnelles est souvent utilisée pour opposer les spécialistes de la fiabilité et de la tolérance aux fautes à ceux de la sécurité, les premiers ayant souvent tendance à négliger les fautes intentionnelles alors que les seconds ne prennent généralement pas en compte les fautes accidentelles. Pourtant, une faute accidentelle arbitraire peut avoir le même effet que n'importe quelle attaque délibérée, et il y a de nombreux exemples de divulgation d'informations confidentielles provoquée par une faute accidentelle ; l'exemple le plus typique est sans doute l'incident survenu au MIT au début des années 70 qui a provoqué l'affichage des noms de tous les utilisateurs avec leurs mots de passe en clair sur tous les terminaux du campus. De même, les techniques de tolérance aux fautes peuvent être appliquées efficacement pour faire face aux intrusions [Rab 89, Des 91], et les techniques de modélisation probabiliste couramment utilisées pour les évaluations prévisionnelles de fiabilité peuvent être étendues pour évaluer la sécurité vis-à-vis des intrusions [Dac 92].

2.3- Erreur et confidentialité

La confidentialité pose un problème particulier vis-à-vis de l'identification des erreurs. Ainsi, une écoute passive (par exemple d'une voie de communication) est à coup sûr une faute intentionnelle malveillante. Il y a défaillance si les spécifications du système précisent qu'il faut empêcher la divulgation des informations qu'a obtenues cette écoute passive. Où est l'*erreur*, au sens qui lui a été donné plus haut : partie de l'état du système susceptible d'entraîner une

---

de protection. Remarquons enfin que si le *principe du moindre privilège* était parfaitement implémenté, l'abus de privilège serait impossible.

défaillance ? L'état du système, au sens strict, n'est pas modifié par l'écoute passive. En revanche, des informations se sont propagées à l'extérieur du système, et donc l'état informationnel de l'environnement a été modifié : l'erreur n'est pas dans l'état du système mais dans l'état de son environnement. Notons que dans cet exemple, il n'y a pas d'erreur latente, l'erreur est générée aux frontières du système, et la défaillance est donc synchrone avec la génération de l'erreur.

Notons cependant qu'en général, pour réaliser une attaque contre la confidentialité, il est nécessaire de modifier l'état du système pour faciliter ou rendre possible l'accès aux informations confidentielles. Il y a dans ce cas d'abord une attaque contre l'intégrité. C'est également le cas des divulgations accidentelles d'informations confidentielles : il faut d'abord qu'il y ait une erreur (provoquée par une faute accidentelle) pour que cette erreur se manifeste sous la forme soit d'une modification des droits d'accès, soit sous la forme d'une sortie des informations confidentielles.

2.4- Sur les étiquettes *faute, erreur* et *défaillance*

L'utilisation exclusive dans ce qui précède des mots, ou plutôt des étiquettes, *faute*, *erreur* et *défaillance* a eu pour but d'illustrer le caractère nécessaire et suffisant de trois notions pour rendre compte des entraves à la sûreté de fonctionnement, ce qui ne préjuge pas de l'utilisation dans des situations particulières soit de synonymes, soit de mots qui désignent, d'une manière condensée et non ambiguë, une classe spécifique d'entrave ; ceci est tout particulièrement vrai pour les fautes (par exemple bogue pour faute logicielle, défaut pour faute attribuable au système de production d'un matériel) et pour les défaillances (par exemple, rupture de service, dysfonctionnement). L'assignation faite ici pour les étiquettes faute, erreur, défaillance tient simplement compte de l'usage : i) évitement des fautes, tolérance aux fautes, diagnostic de faute, ii) détection et correction d'erreur, iii) taux de défaillance. Pour des raisons qu'il n'est pas aisé de distinguer, il y a une réticence certaine à l'utilisation du terme *faute* dans la littérature scientifique et technique française dans le sens où il est utilisé ici : des deux sens communément admis pour la notion de faute, c'est-à-dire manquement à un code moral ou éthique et maladresse, seul le premier sens est généralement retenu, alors que le second est d'un usage tout à fait courant (pour les activités ludiques ou sportives par exemple). Le terme de *panne* lui est généralement préféré. Ce terme, bien que d'un usage courant en Français, est d'une utilisation délicate car il peut désigner tantôt une défaillance (comme dans "tomber en panne"), tantôt une faute (comme dans "trouver la panne") ; de plus, son utilisation devient pour le moins délicate dès que l'on s'écarte des fautes physiques accidentelles : peut-on raisonnablement parler de *panne de conception*, voire de *panne intentionnelle*? L'utilisation de *faute* dans ses diverses acceptions permet de fournir un terme générique qui s'accorde à la généricité de la notion de sûreté de fonctionnement.

3- Les attributs de la sûreté de fonctionnement

Les attributs de la sûreté de fonctionnement ont été définis au § 1 selon diverses propriétés, sur lesquelles on peut mettre un accent plus ou moins prononcé selon l'application à laquelle est destiné le système informatique considéré :

- la disponibilité est toujours requise, bien qu'à des degrés naturellement variables selon les applications ;
- fiabilité, sécurité-innocuité, sécurité-confidentialité[3] peuvent être ou ne pas être requises selon les applications.

---

[3] Nous sommes conscients de l'ambiguïté résultant de l'utilisation du même terme, sécurité, pour désigner deux notions différentes (mais non indépendantes). Nous ne faisons que nous conformer à l'usage établi, tout en notant que réserver, par exemple, sûreté pour la non-occurrence de défaillances catastrophiques, et sécurité pour la prévention d'accès ou de manipulations non-autorisées des informations, ne résoudrait qu'imparfaitement le problème puisque ces deux termes partagent le même adjectif : sûr. C'est ce qui nous a conduits à associer un

Une propriété additionnelle, qui peut être vue comme pré requise pour les autres propriétés, est l'*intégrité*, c'est-à-dire l'intangibilité de l'information, et donc l'évitement de modifications ou suppressions indésirées.

De par leur définition, disponibilité et fiabilité mettent l'accent sur l'évitement des défaillances, la sécurité-innocuité sur l'évitement d'une classe spécifique de défaillances — les défaillances catastrophiques, et la sécurité-confidentialité sur la prévention d'une classe de fautes spécifique — les accès ou manipulations non-autorisées de l'information. Fiabilité et disponibilité sont donc plus proches l'une de l'autre qu'elles ne le sont d'une part de la sécurité-innocuité, et de la sécurité-confidentialité d'autre part. Il est à noter que le qualificatif *confidentialité* a été choisi en accord avec [Gas 88] qui note que la confidentialité est la propriété la plus distinctive de la sécurité, les deux autres propriétés étant l'intégrité et le non-déni de service, c'est-à-dire la disponibilité. Fiabilité et disponibilité pourraient donc être regroupées, et définies collectivement via l'évitement ou la minimisation des interruptions de service. Cependant, cette remarque ne devrait pas conduire à penser que fiabilité et disponibilité sont indépendantes de l'environnement du système : il est connu depuis longtemps que fiabilité et disponibilité d'un *système* sont largement corrélées à son profil d'utilisation, tant vis-à-vis des fautes physiques que des fautes de conception [Iye 82]. Notons enfin que la définition de la sécurité-confidentialité donnée au § 1, la prévention d'accès ou de manipulations non autorisées de l'information, vient de [EEC 91] ; cette référence définit la sécurité comme la combinaison de la confidentialité, prévention d'une divulgation non autorisée de l'information, de l'intégrité, prévention d'une modification non autorisée de l'information, de la disponibilité, prévention d'un déni non autorisé d'accès à l'information ou à des ressources.

Les variations dans l'accent mis sur les attributs de la sûreté de fonctionnement ont une influence directe sur le dosage approprié des techniques décrites aux paragraphes précédents qu'il convient de déterminer pour que le système résultant soit sûr de fonctionnement. Ceci est un problème d'autant plus délicat que certains attributs sont antagonistes (par exemple, disponibilité et sécurité-innocuité, disponibilité et sécurité-confidentialité), d'où la nécessité de faire des compromis. Lorsque l'on considère les trois principales dimensions de la conception d'un système informatique, c'est-à-dire le coût, les performances et la sûreté de fonctionnement, le problème est rendu encore plus délicat par le fait que la dimension de sûreté de fonctionnement est moins bien maîtrisée que les deux autres dimensions [Sie 92].

La *maintenabilité* n'a pas été introduite comme un attribut de base de la sûreté de fonctionnement. La raison en est que, en dépit de sa grande importance pratique, la maintenabilité est plus un moyen d'obtention des autres attributs qu'un but requis pour une application donnée. On peut même dire que la maintenabilité, en tant qu'aptitude aux changements, conditionne les autres attributs de la sûreté de fonctionnement, les changements se rapportant soit à des remplacements de composants défaillants, soit à des corrections de fautes de conception résiduelles, soit encore à des modifications[4].

---

qualificatif pour lever l'ambiguïté : innocuité pour l'évitement des défaillances catastrophiques, confidentialité pour la prévention d'accès ou de manipulations non-autorisées des informations. Naturellement, lorsque le contexte permet de lever aisément le doute, ces qualificatifs ne sont pas nécessaires.

[4] On retrouve ainsi les différentes formes de maintenance qui sont habituellement considérées:

- maintenance corrective, destinée à préserver ou à améliorer l'aptitude du système à délivrer un service en conformité avec la spécification
- maintenance adaptative, qui a pour but d'adapter le système à des modifications de son environnement (par exemple, changement de système d'exploitation ou de système de gestion de bases de données) ;
- maintenance perfective, qui a pour but d'améliorer les fonctions du système, en réponse aux souhaits des utilisateurs — ou des concepteurs, et qui peut comporter l'élimination de fautes dans les spécifications.

Il est à noter que les discussions actuelles sur le bien-fondé de l'emploi du terme maintenance lorsqu'appliqué au logiciel, et plus particulièrement pour les formes de maintenance "adaptative" et "perfective" oublient simplement

La figure 4 résume cette discussion sur les attributs de la sûreté de fonctionnement.

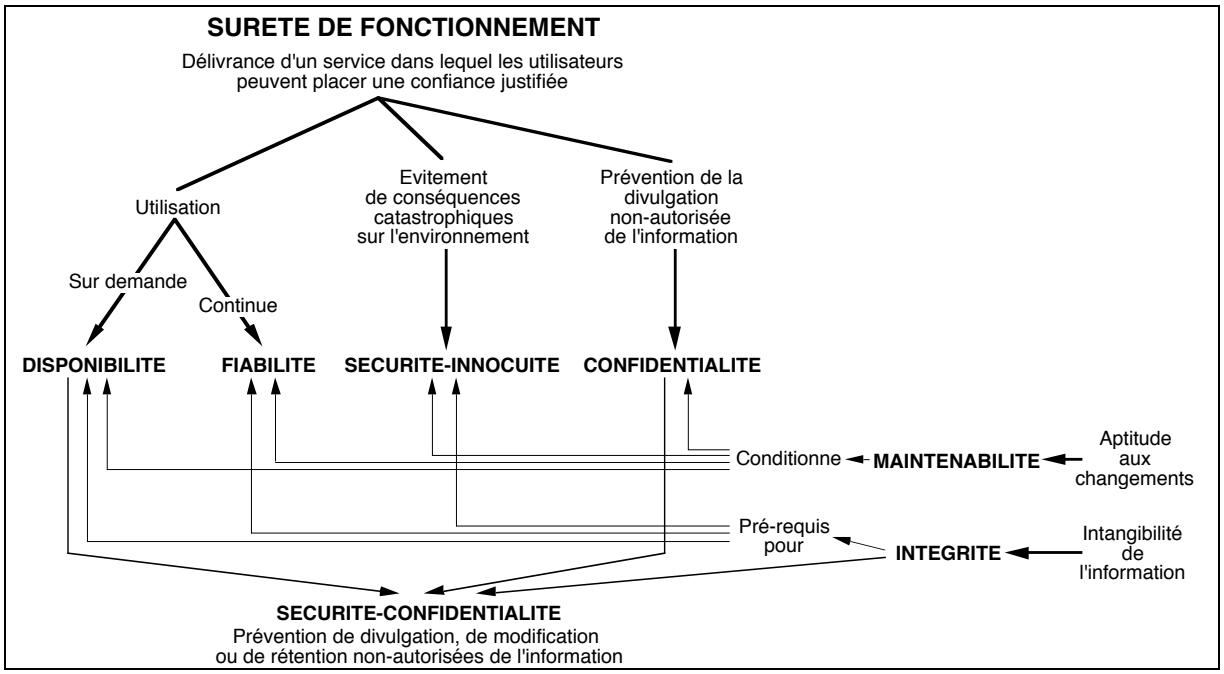

Figure 5 - Relations entre les attributs de la sûreté de fonctionnement

4- Défaillance et spécification

La spécification d'un système décrit ce qui est attendu d'un système en termes a) de sa fonction ou de son service ou des deux, et b) des conditions dans — ou selon — lesquelles la remplir ou le délivrer : environnement, durée, performance, observabilité, … La fonction et/ou le service sont habituellement spécifiés d'abord en termes de ce qui devrait être rempli ou délivré concernant la finalité première du système. Lorsque l'on considère des systèmes de sécurité (soit -innocuité, soit -confidentialité), cette spécification est généralement complétée par l'édiction de ce qui ne devrait pas arriver (par exemple, les états dangereux pouvant mener à une catastrophe, ou la divulgation d'informations sensibles). Cette dernière spécification conduit généralement à spécifier des fonctions ou services additionnels que le système devrait remplir ou délivrer afin de réduire les possibilités de ce qui ne devrait pas arriver (par exemple, authentification d'un utilisateur et vérification de ses droits). Les politiques de sécurité sont de bons exemples de telles spécifications. Notons que si les politiques de sécurité ont d'abord été définies et formalisées pour les systèmes de sécurité-confidentialité, il est parfois possible d'en définir pour les systèmes de sécurité-innocuité [Rus 89]. Ainsi, par exemple, la politique d'intégrité de Biba [Bib 77] peut être étendue pour éviter la propagation d'erreurs dans des systèmes où cohabitent des logiciels de criticités différentes.

Il est essentiel qu'une spécification soit agréée par deux personnes, ou groupes de personnes, physiques ou morales : le fournisseur du système (au sens large du terme : concepteur, constructeur, vendeur, …) et ses utilisateurs humains[5]. L'agrément est nécessaire pour que la spécification serve de base pour déterminer si le service délivré est acceptable ou non.

---

l'étymologie : au Moyen Age, la maintenance désignait les actions nécessaires pour maintenir une armée dans l'état de livrer bataille, donc incluant les deux formes précédemment mentionnées. L'association de maintenance avec réparation de matériel est donc en fait une déviation – récente. Associer "maintenir" à la notion de service permettrait de faire revivre cette signification étymologique, et éliminerait la source même de discussion.

[5] L'agrément peut être implicite, comme lorsque l'on achète un système accompagné de sa spécification et de son manuel d'utilisation, ou lorsque l'on emploie des systèmes "sur étagère".

De plus, ces diverses spécifications peuvent :

a) être exprimées selon divers degrés de détail : spécification des besoins, spécification de conception, spécification de réalisation, …
b) être décomposées selon l'absence ou la présence de défaillance ; le premier cas est relatif à ce qui est habituellement appelé le mode d'opération *nominal*, et le second peut être relatif à un mode d'opération *dégradé*, si les ressources survivantes ne sont plus suffisantes pour assurer le mode nominal.

En conséquence, il n'y a pas généralement une seule spécification, mais plusieurs, et un système peut défaillir par rapport à l'une de ces multiples spécifications, et satisfaire encore les autres. En particulier, les différentes classes d'utilisateurs n'ont pas les mêmes objectifs, et donc ne sont pas concernées de la même façon par les différentes spécifications. Ainsi, un officier de sécurité est d'abord soucieux du respect de la politique de sécurité, alors que la plupart des autres utilisateurs souhaitent la meilleure disponibilité possible. Il se peut même que ces différents objectifs soient contradictoires : par exemple, pour contrer des risques d'inférence dans une base de données, il peut être nécessaire de "brouiller" les résultats de requêtes statistiques, ce qui nuit à la précision de ces résultats.

Par ailleurs, ce qui peut être jugé comme un service acceptable par rapport à une spécification à un niveau de détail donné peut ne pas satisfaire cette spécification à un autre niveau de détail, en raison d'erreurs — de commission ou d'omission — commises en détaillant la spécification, résultant en fait en *fautes de spécification* ; des fautes de spécification peuvent à leur tour affecter n'importe laquelle des diverses spécifications. Plus généralement, une spécification ne peut prétendre être immuable une fois établie. Ce serait simplement ignorance de la vie, qui est *changement*. Les changements peuvent être motivés par la modification de la fonction et/ou du service attendus du système.

Nous sommes donc confrontés à un problème apparemment circulaire : une référence est nécessaire pour déterminer si un service délivré est acceptable ou non, et cette référence peut être fautive. En fait, il importe de reconnaître que si certaines spécifications peuvent être énoncées au début, ou durant, le développement du système, il y a des spécifications qui ne peuvent être déterminées que par l'observation du système dans ses contexte et environnement opérationnels[6]. Cet état de fait se trouve particulièrement vérifié pour les fonctions ou services que le système doit remplir ou délivrer pour réduire les possibilités de ce qui ne devrait pas arriver, donc pour les systèmes de sécurité-innocuité ou de sécurité-confidentialité ; les fautes de spécification sont souvent dans ce cas des fautes d'omission. Les notions de faute, d'erreur, de défaillance s'appliquent aux spécifications, en notant qu'elles concernent alors naturellement d'avantage le *processus* de production du système plutôt que le *produit*, c'est-à-dire le système lui-même. Ceci est renforcé par le fait qu'une faute de spécification peut être consécutive à des décisions non purement d'ordre technique, politiques voire sociales. Enfin, l'importance des fautes de spécification ne doit pas être mésestimée. Bien que pour les systèmes informatiques en général, les défaillances dues à des fautes de spécification soient moins fréquentes durant leur vie opérationnelle que les défaillances dont la cause réside dans les phases de leur développement postérieures à leur spécification (voir par exemple [Gla 81]), la fréquence n'est pas la seule mesure à considérer : la sévérité est également à prendre en compte. De plus, l'argument lié à la fréquence ne s'applique guère dans le cas de systèmes critiques : le soin extrême apporté à leur conception et à leur réalisation fait que les imperfections dans les spécifications constituent généralement une source significative des problèmes rencontrés durant leur vie opérationnelle.

---

[6] La définition usuelle de *spécifier* est "exprimer, déterminer en détail", et ne se réfère donc pas à une précédence entre la chose spécifiée et sa spécification.

Conclusion

Du fait de la généralisation de l'informatique et de sa complexité accrue, l'intérêt et les enjeux de la sûreté de fonctionnement ne sont pas près de s'éteindre : les coûts relatifs à la non-sûreté de fonctionnement des systèmes informatiques (toutes classes de fautes confondues, accidentelles et intentionnelles) sont depuis plusieurs années la première source de sinistres industriels tels que perçus par les compagnies d'assurance, et leur coût excède, en France, les bénéfices de l'ensemble de l'industrie informatique (construction, distribution, services).

Références

Bib 77 K.J. Biba, "*Integrity Considerations for Secure Computer Systems*", MITRE, ESD-TR n.76-372, avril 1977.

Dac 92 M. Dacier, M. Kaaniche, Y. Deswarte, "A Framework For Security Assessment of Insecure Systems," Rapport LAAS n°92 434, 9 pages.

Des 91 Y. Deswarte, L. Blain, J.C. Fabre, "Intrusion tolerance in distributed computing systems", *proc. 1991 IEEE Symposium on Research in Security and Privacy*, Oakland (USA), 20-22 Mai 1991, pp.110-121

EEC 91 Information Technology Security Evaluation Criteria, Provisional Harmonised criteria, Office fort Official Publications of the European Communities, June 1991.

Gas 88 M. Gasser, *Building a Secure Computer System,* Van Nostrand Reinhold, 1988.

Gla 81 R.L. Glass, "Persistent software errors", *IEEE Transactions on Software Engineering,* vol. SE-7, no. 2, March 1981, pp. 162-168.

Iye 82 R.K. Iyer, S.E. Butner, E.J. McCluskey, "A statistical failure/load relationship: results of a multi-computer study", *IEEE Trans. on Computers,* vol. C-31, July 1982, pp. 697-706.

Lap 92 J.C. Laprie (Ed.), *Dependability: Basic Concepts and Terminology*, Springer-Verlag, Vienna, 1992.

Pow 88 D. Powell, G. Bonn, D. Seaton, P. Verissimo, F. Waeselynck, "The Delta-4 approach to dependability in open distributed computing systems", *Proc. 18th IEEE Int. Symp. on Fault Tolerant Computing (FTCS-18),* Tokyo, Japan, June 1988, pp. 246-251.

Rab 89 M.O. Rabin, "Efficient dispersal of information for security, load balancing and fault tolerance", *Jounal of the ACM*, vol. 36, no. 2, April 1989, pp. 335-348.

Rus 89 J. Rushby, "Kernels for safety?", in *Safe and Secure Computing Systems*, T. Anderson, Ed., Blackwell Scientific Publications, 1989, pp. 210-220.

Sie 92 D.P. Siewiorek, D. Johnson, "A design methodology for high reliability systems: the Intel 432", in D.P. Siewiorek, R.S. Swarz, *Reliable Computer Systems, Design and Evaluation*, Digital Press, 1992, pp. 737-767.