Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Discipline ou spécialité :

Systèmes Industriels

Présentée et soutenue par :
Mickaël DIEVART

le : vendredi 3 décembre 2010

Titre :

Architectures de Diagnostic et de Pronostic Distribuées de Systèmes Techniques Complexes de Grande Dimension

---

JURY

M. Noureddine ZERHOUNI, Professeur des Universités à l'ENS2M de Besançon
Mme Mireille BAYART, Professeur des Universités à l'Université de Lille1, Polytech-Lille
Mme Zineb SIMEU-ABAZI, Maître de Conférences, HDR au Laboratoire G-SCOP de Grenoble
M. Philippe CHARBONNAUD, Professeur des Universités au LGP de l'ENI de Tarbes
M. Xavier DESFORGES, Maître de Conférences au LGP de l'ENI de Tarbes

---

Ecole doctorale :
Systèmes (EDSYS)

Unité de recherche :
Laboratoire Génie de Production (LGP), Ecole Nationale d'Ingénieurs de Tarbes (ENIT)

Directeur(s) de Thèse :

M. Philippe CHARBONNAUD, Professeur des Universités à l'ENI de Tarbes

Rapporteurs :
M. Noureddine ZERHOUNI, Professeur des Universités à l'ENS2M de Besançon
Mme Mireille BAYART, Professeur des Universités à l'Université de Lille1, Polytech-Lille
Mme Zineb SIMEU-ABAZI, Maître de Conférences, HDR au Laboratoire G-SCOP de Grenoble

Remerciements

C'est avec un grand plaisir que je remercie les personnes qui, de près ou de loin, ont contribué à la réalisation de cette thèse.

Je remercie très sincèrement Monsieur Daniel NOYES, de m'avoir accueilli au sein du Laboratoire Génie de Production (LGP) de l'Ecole Nationale d'Ingénieurs de Tarbes (ENIT).

Je remercie les membres du jury de cette thèse et tout particulièrement les rapporteurs : M. Noureddine ZERHOUNI, Professeur des Universités à l'ENS2M de Besançon, Mme Mireille BAYART, Professeur des Universités au LAGIS de Polytech' Lille de Villeneuve d'Ascq, Mme Zineb SIMEU-ABAZI, Maitre de Conférences et HDR du Laboratoire G-SCOP de l'INP de Grenoble qui ont accepté d'examiner cette thèse dans des délais courts.

Je remercie M. Philippe CHARBONNAUD, Professeur des Universités au LGP de L'ENI de Tarbes, de m'avoir proposé cette thèse et accueilli au sein de son équipe ainsi que M. Xavier DESFORGES, Maître de Conférences au LGP de L'ENI de Tarbes. Je les remercie pour leur suivi et participation à mon travail ainsi qu'à la rédaction d'articles. Ils ont su à la fois diriger et conseiller mon travail tout en me laissant une grande liberté dans mes investigations et choix de réalisation. Je tiens aussi à les remercier pour nos nombreuses discussions et leurs précieux conseils lors de la rédaction de ce mémoire. J'espère fortement que nos collaborations pourront se poursuivre à l'avenir.

Je remercie aussi tous les membres du LGP pour l'agréable ambiance de travail aussi bien dans le cadre de la recherche que de l'enseignement.

Sommaire

Liste des tableaux

Liste des algorithmes

Liste des figures

Introduction générale

Problématique et objectifs

Les grands systèmes sont caractérisés par un grand nombre de variables, des non linéarités et des incertitudes. Leur décomposition en sous-systèmes, plus facilement gérables et organisés éventuellement de façon hiérarchique, est dépendante d'échanges importants d'informations parfois même en temps critique mais est aussi dépendante des mécanismes de décentralisation ou de coordination. Aussi de nouveaux développements sont-ils nécessaires pour résoudre les problèmes liés à la taille et à la complexité de tels systèmes. Ils sont relatifs aux méthodologies et aux outils pour traiter de leur modélisation, de leur conception, de leur conduite, ou encore de l'aide à la décision.

Les systèmes industriels recouvrent de nombreuses formes. Aujourd'hui, ils sont le plus souvent organisés en réseaux. Les nouvelles technologies de l'information et de la communication apportent un ensemble de moyens supplémentaires pour réaliser des applications ayant un intérêt majeur pour renforcer l'exploitation sûre de ces systèmes et la sécurité des personnes..

Dans ce mémoire, nous abordons une sous catégorie importante des systèmes industriels : les systèmes techniques pour lesquels l'homme n'est pas assimilable à un composant du système, i.e. la faute humaine ne fera pas l'objet d'un diagnostic, l'erreur de programmation sera également hors du champ d'investigation. Du fait de l'évolution des architectures des systèmes, nous avons étudié le diagnostic et le pronostic distribués de Systèmes Techniques Complexes de Grande Dimension (STCGD).

Souvent, il sera fait référence à des systèmes embarqués pour illustrer la présentation. Cela vient du fait que cette thèse s'est déroulée dans le cadre d'un contrat entre Airbus France et l'ENIT. Il est cependant normal de s'intéresser à ce type de systèmes car il assure le transport de biens ou de personnes dont l'intégrité à une valeur très importante.

Les systèmes de transport (voitures, camions, métros, trains, avions, etc.) sont

composés de calculateurs, câblage, capteurs, logiciels... Ces composants sont regroupés en modules, entités remplaçables [1], et permettent d'assurer une fonction précise. Lors de la défaillance d'un équipement, l'opérateur de maintenance doit pouvoir remplacer ou réparer les entités défaillantes et uniquement celles-ci. Pour cela, il peut disposer de moyens de diagnostic, le plus souvent embarqués, et peut avoir à pratiquer des séries de tests lui permettant d'isoler les modules défaillants dans un réseau d'instruments hétérogènes. Le système de surveillance ayant pour objectif, à partir des intrants que sont la mission, les opérateurs sur système et les opérateurs de maintenance, de fournir des alarmes. Le diagnostic doit fournir quant à lui des listes de composants (et/ou de fonctions) défaillants alors que le système est soumis à un environnement changeant, à des perturbations ou à une utilisation différente suivant l'opérateur. Dans le cas de dégradations, le système peut alors sortir de son mode de fonctionnement normal et donc ne plus pouvoir assurer sa mission selon des critères de performance et de sécurité pour lesquels il a été conçu.

Il est usuel de distinguer les traitements effectués en ligne, réalisés durant l'engagement du système, des traitements hors lignes, i.e. hors engagement. Pour un avion, les traitements en ligne correspondent à des élaborations d'informations effectuées pendant le vol et, pour les traitements hors lignes, à des élaborations d'informations effectuées lors de l'escale. Durant l'engagement du système, seul des traitements de données, de capitalisation de données et quelques tests n'influençant en rien les règles de sécurité préconisées peuvent être effectués. Si une défaillance d'une fonction du système survient, une reconfiguration et/ou une marche en mode dégradé peuvent être envisagées. C'est seulement après la phase d'engagement que les techniciens de maintenance pourront procéder à tous les tests et remplacer ou réparer le(s) module(s) dont le dysfonctionnement a entraîné la défaillance. Il est à noter que cette phase peut éventuellement être abrégée ou les objectifs peuvent être révisés selon la criticité de la fonction. Les techniciens peuvent néanmoins se baser sur les résultats produits par les traitements embarqués de détection de fautes et de diagnostic en ligne et ceux effectués hors ligne permettant de raccourcir la durée de la recherche du ou des modules à l'origine de la défaillance. Cela permet une intervention de maintenance plus rapide contribuant ainsi à l'amélioration de la disponibilité du système tout en restant dans les normes de sécurité préconisées.

Cependant, les traitements embarqués pour la détection d'erreurs et le diag-

1. On peut dire également réparable. Usuellement, pour des raisons de disponibilité du système, remplacer « signifie » réparer. Dans la pratique, certains composants sont parfois réparés et non pas remplacés. Les composants remplacés peuvent être effectivement réparés par la suite en étant considérés à leur tour comme des systèmes.

nostic de fautes permettant d'identifier les dysfonctionnements de modules et la défaillance de fonctions ne sont pas pleinement pertinent vis-à-vis d'une politique de maintenance préventive conditionnelle, même s'ils permettent d'améliorer les performances d'actions de maintenance corrective. En effet, ils ne facilitent que la constatation des dysfonctionnements et des défaillances lorsque ceux-ci ont eu lieu. Ainsi, une maintenance préventive conditionnelle rentre d'avantage dans un processus proactif, où la prévention de la défaillance est recherchée, alors que la maintenance corrective est purement réactive car mise en place après avoir subit la défaillance. La maintenance préventive conditionnelle est basée sur la surveillance de l'évolution de grandeurs caractéristiques relatives à la santé du composant et à des traitements de pronostic à partir de ces grandeurs. La maintenance préventive conditionnelle, de ce point de vue, assure donc un meilleur engagement des composants par rapport à la maintenance préventive systématique car elle permet une utilisation plus optimale des composants.

Les systèmes embarqués sont devenus de plus en plus complexes pour répondre à la fois à des réglementations toujours plus exigeantes en termes de protection de l'environnement, de sécurité des biens et des personnes et à des besoins en termes de nouveaux services. Ces réglementations et besoins ont ainsi contribué à l'introduction de nouvelles fonctionnalités accroissant la complexité des systèmes embarqués. Pour faire face à cette complexité croissante, les multiples fonctionnalités des systèmes embarqués ont été structurées en réseaux de fonctions plus élémentaires supportées par des composants modulaires facilement remplaçables. Ces composants émanent de différents fournisseurs proposant diverses options à leurs produits. Cette variété de fournisseurs et cette diversité de variantes rendent plus ardu l'accès à la connaissance de conception et à leur différenciation après intégration dans un STCGD. Les configurations sont multiples à l'assemblage de ces différents ensembles et engendrent des problèmes d'intégration. L'augmentation du nombre de fonctionnalités des systèmes embarqués a contribué à en augmenter les coûts d'acquisition et de possession conduisant ainsi les exploitants de ces systèmes à chercher une optimisation de leur disponibilité pour obtenir un meilleur taux d'engagement.

Notre objectif principal est lié à l'amélioration de la disponibilité des STCGD. Pour cela, nous proposons des techniques et des outils d'aide au diagnostic technique plus efficients pour cette catégorie de systèmes.

Les objectifs intermédiaires portent sur les façons d'implanter une fonction diagnostic au sein même du STCGD et sur son efficacité. L'objectif est d'éviter l'écueil des architectures centralisées qui donnent une grande quantité d'informa-

tions à traiter pouvant être parfois erronées. Ces erreurs combinées à la suite des nombreux traitements effectués par la fonction de diagnostic conduisent à la dépose à tort de composants remplaçables, induisant des surcoûts de maintenance et des risques de dégradation du système lors de ces interventions de remplacement inutiles. Afin de réduire de tels risques, les architectures décentralisées de diagnostic constituent une voie d'investigation logique. Car ainsi le diagnostic peut être effectué au plus près des composants. Les traitements locaux assurent la surveillance des composants et leur diagnostic. Pour ce type d'architecture, il faut proposer une technique pour assurer que les traitements locaux permettent de diminuer le nombre de composants incriminés vers le nombre de composants effectivement en faute. Cependant, tous les traitements ne peuvent pas être effectués lors de l'engagement du système principalement pour des raisons de sécurité. Subir la défaillance est la situation à proscrire car elle ne permet pas de planifier les interventions de maintenance. Des nouvelles politiques basées sur des méthodes fiabilistes permettent de devancer fréquemment la défaillance de l'élément et facilitent ainsi la planification des opérations de maintenance. La maintenance préventive systématique ne permet cependant pas une utilisation de l'équipement sur sa durée de vie complète. Elle ne permet pas non plus d'annuler complètement le risque d'occurrences de défaillances d'équipements, modules ou composants avant l'échéance prévue, d'où la nécessité de pouvoir surveiller les paramètres significatifs de dégradation d'un équipement lorsque cela est possible. Aussi est-il nécessaire de fournir des outils permettant d'implanter des traitements de détection d'erreurs, de diagnostic de fautes et de pronostic adaptés aux STCGD et ainsi de fournir une aide pour définir une utilisation optimale de l'équipement tout en limitant les risques de défaillance.

Contribution de la thèse

Dans cette thèse, la contribution principale porte sur la proposition d'une architecture distribuée pour le diagnostic et le pronostic de STCGD permettant de définir l'état de santé de ce type de système.

Un STCGD peut être défini comme un système mettant en œuvre différents systèmes de multiples fournisseurs qui interagissent entre eux dans un but commun et devant, la plupart du temps, répondre à des critères de performances, de fiabilité et de sécurité prédéfinis (Murthy et Krishnamurthy, 2009) . Chacun de ces systèmes est assuré par un certain nombre de sous-systèmes eux-mêmes implémentant des fonctions. Ces fonctions sont elles mêmes assurées par un ensemble de composants.

L'une des évolutions majeures des « systémiers » vise à doter leurs produits d'un service d'évaluation de l'état de santé du système fiable qui doit satisfaire ses clients. C'est un important challenge pour lequel des évaluations d'architectures ont été faites dans différents domaines. Nous en présentons les éléments marquants. Pour le domaine d'application des systèmes de transport, nous avons réalisé une étude comparative (Dievart et Charbonnaud, 2007). Elle a permis d'identifier la problématique et de justifier notre démarche.

Généralement, l'évaluation de l'état de santé est élaborée à partir des données issues des fonctions de diagnostic et de pronostic (Vachtsevanos et Wang, 2001). L'ajout d'une fonction de pronostic est justifié également par la préoccupation des exploitants à s'assurer de la capacité de leurs systèmes à être engagés dans des conditions de fiabilité acceptable. C'est pourquoi nous avons défini les principes d'évaluation distribuée de l'état de santé des STCGD (Dievart *et al.*, 2008). Durant sa vie, un système est sujet à diverses défaillances ayant un impact sur son mode de fonctionnement. Lors de l'apparition d'une défaillance, si celle-ci entraîne la défaillance totale du système, la défaillance est qualifiée de cataleptique. La défaillance peut être simple, multiple ou cachée. Une défaillance dite de dysfonctionnement contraint le système à opérer en mode dégradé impliquant donc une baisse de performance. La dégradation de performance peut affecter localement une fonction ou globalement le système. Lorsque la défaillance est dite fugitive, le système rentre dans un mode intermittent, i.e., il alterne état normal de fonctionnement et état anormal.

Pour évaluer l'état de santé de ces systèmes, il est nécessaire d'en avoir une connaissance la plus complète. Parmi les diverses méthodes de modélisation, celles isolant les connaissances issues des études menées durant la phase de conception du système paraissent pertinentes. A cela peut s'ajouter le retour d'expérience qui permet d'affiner la connaissance a priori que l'on avait du système. Notre approche du diagnostic est basée sur quatre types de connaissances : fonctionnelle, structurelle, comportementale et topologique (Dievart *et al.*, 2009). L'analyse fonctionnelle recense les fonctions du système physique qui possèdent des indicateurs pertinents et significatifs. Ces fonctions correspondent à l'analyse du système physique, aux données de conception et à l'analyse experte. L'analyse structurelle s'applique aux fonctions identifiées en recensant l'ensemble des composants et les connexions physiques entre eux. L'analyse comportementale permet de recenser les variables et leurs relations (relation de causes à effet, de causalité). L'analyse topologique permet de déterminer les proximités entre les composants pouvant entraîner une défaillance indirecte.

Différents types d'architectures permettant de déployer ces fonctions de surveillance peuvent être considérés lors de la conception d'un système. Cependant, avantages et inconvénients méritent d'être comparés pour chacune de ces architectures (Zennir, 2004). Dans une architecture centralisée de diagnostic un seul et unique organe est chargé d'établir un diagnostic, ce qui pose évidemment des problèmes de fiabilité. Pour éviter le problème de charge de l'organe principal d'une architecture centralisée et ainsi pouvoir surveiller des systèmes fournissant des quantités d'informations supérieures, les architectures décentralisées peuvent être envisagées (Pencolé et Cordier, 2005; Console *et al.*, 2007). Elles permettent de diminuer la charge due aux masses de données en la distribuant via des agents de diagnostic locaux. Ceux-ci envoient ensuite leurs résultats à un agent de diagnostic global qui assure le diagnostic de l'ensemble du système. Mettre en place une approche distribuée doit permettre d'améliorer la fiabilité du système. En effet, aucun organe n'est hiérarchiquement en charge d'établir un diagnostic global du système, ce qui implique qu'un diagnostic global existe toujours à moins que tous les agents locaux tombent en panne. Les agents locaux de diagnostic doivent communiquer entre eux afin de tendre localement et globalement à un diagnostic cohérent (Ardissono *et al.*, 2005; Biteus *et al.*, 2008). Nous avons spécifié une plateforme pour évaluer les approches centralisées, décentralisées et distribuées (Dievart *et al.*, 2010b). Le diagnostic et le pronostic ont été implémentés et des critères de performances ont été proposés pour permettre une comparaison. L'évaluation des performances de différentes architectures centralisées, décentralisées et distribuées de diagnostic et de pronostic a été réalisée pour différents systèmes (Dievart *et al.*, 2010a).

Organisation du mémoire

Ce document est organisé en quatre chapitres : Le chapitre 1 délimite le périmètre des STCGD et résume les moyens de surveillance disponible. Une définition des STCGD est proposée et leurs moyens de surveillance lorsqu'ils sont embarqués sont présentés. Les différentes architectures pour le contrôle et la surveillance des STCGD sont discutées ainsi qu'une proposition de modélisation de ces systèmes. Ce qui peut être attendu comme une aide à la maintenance conditionnelle et à la planification des systèmes est résumé. Les problématiques de maintenance conditionnelle et d'évaluation de l'état de santé sont alors définies. Les types de diagnostic et de pronostic sont présentés afin d'aboutir à une évaluation de l'état de santé des STCGD. Les études relatives au diagnostic décentralisé sont discu-

tées puis les apports des NTIC et des technologies distribuées au diagnostic sont présentés. Par la suite, le diagnostic distribué et les travaux relatifs à ce mode de déploiement sont introduits. Les travaux proposant une architecture distribuée de la fonction de pronostic pour les STCGD sont quasi inexistants, comme cela est montré par une étude bibliographique. L'objectif majeur de ce premier chapitre reste une présentation des problématiques abordées relatives, principalement, à l'évaluation de l'état de santé d'un système et sur les différentes architectures applicatives. Les limites des approches centralisées et décentralisées du diagnostic sont présentées et confrontées à l'apport des approches distribuées. Finalement, la conclusion rappelle la nécessité de modéliser les STCGD et les connaissances issues de la conception pour être utilisées par les fonctions de diagnostic et de pronostic.

Le chapitre 2 est dédié à l'analyse des fonctions de diagnostic et de pronostic dans le cas des STCGD. Il a pour but de définir les entrées de ces deux fonctions dont certaines sont produites par la fonction de surveillance. Les résultats produits par les fonctions de diagnostic et de pronostic sont également présentés et formalisés. Les informations et/ou les connaissances supports aux traitements réalisés par ces fonctions ainsi que leur modélisation afin de les exploiter sont elles aussi décrites et formalisées. Cette connaissance constitue l'information minimale à l'évaluation de l'état de santé d'un système. Finalement, une caractérisation des statuts que peut prendre un composant est proposée. Couplée à la connaissance comportementale, elle est principalement utile à la fonction de diagnostic pour définir le mode de fonctionnement d'un composant.

Le chapitre 3 présente les principes de diagnostic et de pronostic nécessaires à l'évaluation de l'état de santé des STCGD ainsi que les données échangées. Le projet OSA-CBM définit une couche d'évaluation de santé (Health Assessment - HA) dont la fonction principale est de déterminer l'état de santé d'un système, d'un équipement ou d'un composant surveillé. Les différents résultats fournis par les fonctions de diagnostic et de pronostic doivent être considérés comme une aide à la maintenance pour prendre les décisions adéquates pour maintenir le système, comme cela est considéré dans la couche « Decision Support » d'OSA-CBM. Dans notre approche, le Health Management (HM) est introduit comme étant une combinaison des résultats des couches de diagnostic et de pronostic. L'évaluation de l'état de santé d'un STCGD passe par une évaluation de l'état de santé actuel du système, fournie par la fonction de diagnostic, et une évaluation de la santé future du système, fournie par la fonction de pronostic. Ce chapitre décrit les prérequis nécessaires pour la couche de surveillance des STCGD et les principes du diagnostic et du pronostic sont ensuite présentés sous la forme de différents

algorithmes. Enfin, une méthode d'évaluation de l'état de santé des STCGD est proposée.

Le chapitre 4 sert de cadre au déploiement des principes des fonctions permettant l'évaluation de la santé des STCGD. Plusieurs déploiements peuvent être envisagés, il est paru intéressant de proposer une plateforme pour en évaluer leurs performances. Notre première proposition a été celle d'un déploiement distribué. Il a fait l'objet d'une comparaison de performances avec un déploiement centralisé. Pour effectuer ces comparaisons, la plateforme de simulation a eu pour but de se comporter comme la couche de surveillance d'un STCGD. Un cas d'étude paramétrable est proposé pour chacun des deux déploiements et leurs performances sont comparées.

Nous concluons en établissant l'ensemble des apports de cette thèse avant d'établir les perspectives de travail permettant de faire évoluer les algorithmes de diagnostic et de pronostic ainsi que la plateforme sur laquelle ils sont déployés.

Chapitre 1

Diagnostic et pronostic de systèmes techniques complexes de grande dimension

1.1. Introduction

Dans ce chapitre, nous présentons un Etat de l'Art du diagnostic et du pronostic après avoir introduit la classe des Systèmes Techniques Complexes de Grande Dimension (STCGD).

Ainsi, nous délimitons la classe des STCGD et présentons les moyens de surveillance auxquels ils font appels. Nous décrivons les particularités de ce type de système lorsqu'ils sont embarqués. Les différentes architectures pour le contrôle et la surveillance des STCGD sont présentées ainsi qu'une proposition de modélisation. Nous exprimons les attentes d'une aide à la maintenance conditionnelle et à la planification des systèmes. La problématique de la maintenance conditionnelle et de l'évaluation de l'état de santé est alors présentée. Les types de diagnostic et de pronostic sont présentés afin d'aboutir à une évaluation de l'état de santé des STCGD. Nous rappelons les formes de pronostic (basées sur les modèles, guidées par les données et basées sur l'expérience) et précisons leurs apports pour l'aide à la maintenance. Puis nous introduisons une discussion sur le diagnostic décentralisé en le confrontant au diagnostic distribué. Nous présentons les études relatives au diagnostic décentralisé puis établissons les apports des NTIC et des technologies distribuées au diagnostic. Concernant le pronostic, les travaux proposant une architecture distribuée de cette fonction pour les STCGD sont quasi inexistants, d'après l'étude bibliographique présentée. L'objectif majeur de ce premier chapitre reste la mise en évidence des problématiques abordées portant principalement sur l'évaluation de l'état de santé d'un système et sur les différentes architectures permettant de déployer les fonctions permettant d'y arriver. Les limites des approches centralisées et décentralisées du diagnostic y sont discutées et confrontées à l'apport des approches distribuées.

Finalement, nous concluons sur la nécessité de modélisation des STCGD et des connaissances issues de la conception pour être utilisées par les fonctions de

diagnostic et de pronostic. La formalisation et la définition des entrées et sorties de ces fonctions doivent aussi être entreprises.

1.2. STCGD et surveillance

1.2.1. Définition des STCGD

Les systèmes techniques complexes sont présents dans la vie courante et ceci de manière plus ou moins visible. Composés de multiples briques technologiques en interaction éventuelle avec des opérateurs ou des utilisateurs, ils prennent en compte de nombreuses informations pour réaliser une opération complexe de façon plus ou moins automatique. La complexité du système est généralement transparente pour l'utilisateur final, mais ses défaillances peuvent avoir de lourdes conséquences humaines, sociales ou économiques.

Un système technique complexe est composé de sous-systèmes, dont la conception et le fonctionnement font intervenir différents corps de métiers qu'aucun d'entre-eux ne pourrait appréhender dans son ensemble. Un système complexe est un système composé de différents composants interconnectés qui présente, dans son ensemble, différentes propriétés pas nécessairement issues de chacunes des propriétés des composants le constituant (Murthy et Krishnamurthy, 2009).

La difficulté dans les systèmes complexes réside donc dans la manière de les appréhender dans leur ensemble et non pas par les composants les constituant. La conception d'un système complexe requiert donc des méthodes et des outils garantissant la conformité des composants des sous-systèmes et du système fini aux spécifications tout au long de leur réalisation (qualité de services, capacité d'accueil de nouveaux sous-systèmes, etc.). Le verrou essentiel dans la conception des systèmes complexes demeure leur fiabilité et leur sûreté de fonctionnement (Zolghadri, 2002). Cette fiabilité doit être attestée par construction grâce aux méthodes de conception mises en place. Lors de la conception d'un système complexe, l'architecture du système doit être adaptée à la fonction recherchée et au degré de fiabilité et de sûreté nécessaire (y compris par l'introduction de redondances), tout en tenant compte de la durée de vie du système. De plus, la diversité des technologies et des compétences liées mises en œuvre doit être prise en compte dès la conception, ce qui suppose des outils génériques assurant les liens entre plusieurs de ces technologies et/ou de ces compétences et des modélisations multidomaines du même système (Saranga et Knezevic, 2001).

Un système complexe peut avoir besoin de tolérances aux fautes, comme devoir assurer un service minimal dans des conditions de fonctionnement altérées par une

reconfiguration du système. L'arrivée des Nouvelles Technologies de l'Information et de la Communication (NTIC) a permis l'implémentation d'architecture ouverte (par exemple SOA - Service Oriented Architecture) rendant possible la détection automatique des éléments qui composent le système et des services qu'ils offrent assurant ainsi l'indépendance du système vis-à-vis des fournisseurs d'équipements (Haverkamp et Richards, 2002).

Un STCGD peut être défini comme un système mettant en œuvre différents systèmes de multiples fournisseurs qui interagissent entre eux dans un but commun et devant la plupart du temps, répondre à des critères de performances, de fiabilité et de sécurité prédéfinis (Murthy et Krishnamurthy, 2009).

1.2.2. Surveillance des STCGD embarqués

Afin d'assurer un suivi du processus et de détecter toutes défaillances intervenant sur ces systèmes, il est nécessaire d'instrumenter le processus à des fins de surveillance pour établir un diagnostic et ainsi de conseiller les techniciens de maintenance. Un capteur "intelligent" réalise une transduction, un conditionnement, un traitement numérique et a une interface rendant possible la communication avec un bus bidirectionnel numérique. Les réseaux de terrain assurent une interconnexion des différents modules constituant le système (CIAME, 1999; Heinzelman *et al.*, 2004; Alex *et al.*, 2004; Henricksen et Robinson, 2006). D'un point de vue applicatif, les capteurs intelligents font d'ores et déjà partie intégrante des grands systèmes complexes comme les avions, les véhicules spatiaux (Barnhart et Ziemer, 1991; Hedley *et al.*, 2003) ou les automobiles, les bâtiments, les ouvrages d'art ou encore les machines-outils. De plus, ils ont une capacité à gérer un historique et sont aussi très facilement configurables (changement aisé des seuils de détection, ajout de nouveaux services liés à la mesure) (Robert *et al.*, 1993; Desforges et Archimède, 2006). Ce type de capteurs se révèle efficace pour diagnostiquer des systèmes de plus en plus complexes dont le comportement est difficilement prévisible selon les différents modes de fonctionnement de ses sous-systèmes. De plus, les actionneurs intelligents peuvent également servir à la conduite et à la surveillance du système (Staroswiecki et Bayart, 1994). Grâce à leur capacité à s'autodiagnostiquer, il est possible de diagnostiquer le système lui-même, mais aussi sa fonction de surveillance. Toutefois, il existe peu d'architectures capables de gérer cet important flot de données hétérogènes.

Un système embarqué est un système constitué d'un microprocesseur implémenté pour assurer une ou plusieurs fonctions (Heath, 2003). L'ingénierie des systèmes embarqués fait référence aux méthodes, techniques et outils (équipements,

logiciels, plates-formes) pour la conception et le développement de sous-systèmes intelligents capables de contrôler une large gamme d'équipements électroniques, de systèmes industriels, d'infrastructures. Les systèmes embarqués peuvent être vus comme des ordinateurs enfouis dans les équipements électroniques du quotidien (téléphones, voitures, avions, satellites, engins et équipements industriels) (Biteus *et al.*, 2005). Le logiciel occupe une place de plus en plus importante dans les systèmes embarqués si bien que certaines fonctions qui étaient assurées par des biens matériels sont remplacées par du logiciel. Ainsi, la frontière entre le matériel, répondant à des questions de performance et le logiciel, utilisé pour sa flexibilité, est de plus en plus imperceptible. L'utilisation de la co-conception matérielle-logicielle n'étant pas encore généralisée, les techniques usuelles de conception de systèmes embarqués conduisent à spécifier de façon disjointe l'aspect matériel et l'aspect logiciel. L'acceptabilité des fonctions logicielles, leur utilisation et leur généralisation sont donc en jeu.

La généralisation des systèmes embarqués dans l'environnement quotidien tend à prendre appui sur les capacités des systèmes hétérogènes à communiquer et à s'adapter aux changements de contexte. De fait, des systèmes embarqués provenant de divers équipements et de diverses origines doivent pouvoir interopérer. Cela induit la nécessité de disposer de standards ouverts prenant en compte les spécificités des équipements embarqués.

Dans les systèmes critiques (avionique, nucléaire, spatial, etc.), les comportements doivent être garantis car des défaillances peuvent avoir des conséquences irréversibles. Ils sont donc soumis à diverses contraintes dont certaines sont critiques (Ibrahim, 2004) le coût, l'espace compté, la puissance de calcul, la consommation énergétique faible, le temps réel et, enfin, la sûreté. Dans le développement de systèmes moins critiques des compromis doivent être trouvés entre plusieurs critères comme la qualité de service, les coûts, les délais de conception et de déploiement, la consommation d'énergie des équipements ou bien encore la sécurité.

Pour les grands intégrateurs, l'ingénierie des systèmes embarqués impacte directement les coûts et délais de développement. Près de la moitié des projets de développement des systèmes embarqués sont lancés avec plusieurs mois de retard, et moins de la moitié de toutes les conceptions atteignent leurs objectifs de départ en matière de performances. Ces difficultés sont principalement dues à la complexité de la réalisation de tels systèmes (Meinadier, 1998).

La surveillance des systèmes embarqués peut être déployée sur un bus. Dans le domaine aéronautique, le bus ARINC 429 faisait jusqu'alors office de standard dans les architectures avioniques. Ce bus est de type mono-émetteur et peut supporter

jusqu'à 20 récepteurs et ne permet que des débits de 12 kbits/s ou 100 kbits/s. Dans l'automobile, la norme ISO 9944 spécifie un bus dédié au diagnostic des calculateurs embarqués assurant des débits pouvant aller jusqu'à 19 kbits/s semblable au bus LIN où chacun des abonnés se comporte aussi bien en émetteur qu'en récepteur mais un seul, le maître, définie la cadence des échanges. Ce n'est qu'à partir des années 1990 que sont apparus des bus autorisant des communications bidirectionnelles et beaucoup plus rapide comme le bus CAN dans l'automobile ou bien encore le bus ARINC 629 dans l'aéronautique (Berger, 1997; Audsley et Grigg, 1997). La limitation du nombre de récepteurs implique la mise en place d'une multitude de bus de communication sur un aéronef (368 dans les avions Airbus A340). Ces bus sont en général de nature différente et imposent donc l'utilisation de matériels permettant de faire transiter une information sur ces différents supports de l'émetteur à son destinataire (module d'entrées/sorties). Plus récemment, des bus sont apparus autorisant des débits de 100Mbits/s comme le bus AFDX (Ethernet Full Duplex certifié avionique) mis en place sur l'A380 (Grieu, 2004; Mifdaoui, 2007). Cette technologie doit répondre à des contraintes de bande passante et de déterminisme dans les communications pour pouvoir répondre à des contraintes imposées par des certifications.

L'une des grandes exigences actuelles dans la maintenance des systèmes consiste à ne plus subir les défaillances des composants mais à les prédire. L'objectif étant d'utiliser les équipements au maximum de leur durée de vie pour les rentabiliser et de pouvoir prévoir le matériel nécessaire à leur remplacement une fois la mission terminée. Cela permet d'augmenter la disponibilité du système sans impacter la sécurité des utilisateurs (opérateurs ou clients). Dans le cas de la maintenance des avions, certains équipements nécessitent l'utilisation d'outils spécialisés de dépose de composants n'étant disponibles que dans le local de maintenance principal où se situe le siège de la MRO (Maintenance, Repair and Overhaul) de la compagnie exploitant l'avion. Ainsi, différents systèmes provenant de différents équipementiers sont embarqués et communiquent entre eux afin de fournir un ou des services précis à son utilisateur dont certains dans des performances devant répondre à un cahier des charges strict.

L'évolution des techniques et la miniaturisation des composants permettent donc à chacun de ces systèmes d'embarquer plus de décision localement. L'hétérogénéité due à la quantité et au format des informations émises par les systèmes de ces différents équipements rend la tâche de surveillance encore plus compliquée. Il faut alors spécifier le format et le contenu des informations fournies par les équi-

pements dans un but de surveillance et de maintenance du système (Bengtsson *et al.*, 2004).

1.2.3. STCGD et leur modélisation

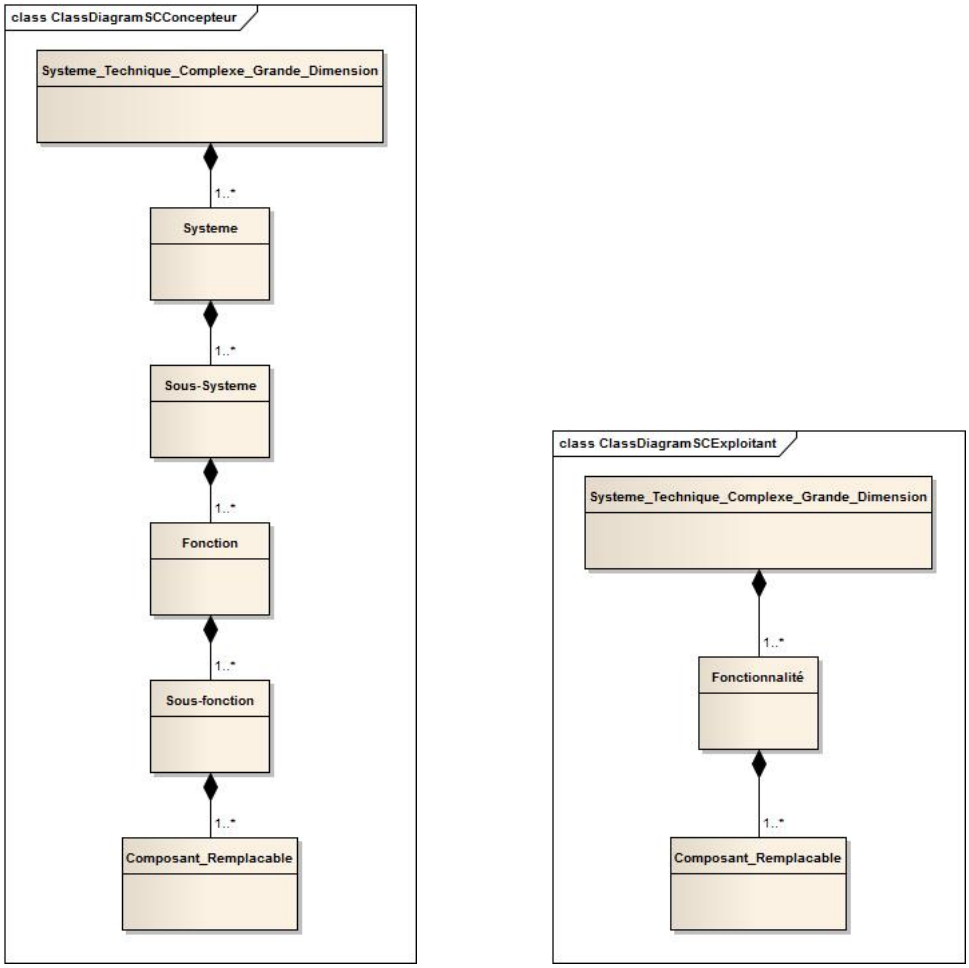
D'un point de vue conception, un STCGD est réalisé par un certain nombre de sous-systèmes eux-mêmes implémentant des fonctions. Ces fonctions sont elles mêmes assurées par l'interconnexion de Composants Remplaçables (ou réparables) (CR). Ce découpage est utile lors de la phase de conception du système et est conservé ensuite car il répond à un cahier des charges bien spécifique et à des performances pré-établies issues de contraintes permettant à ces systèmes d'obtenir des certifications leurs permettant d'être exploités. Ceci est le cas notamment dans les domaines aéronautiques et spatiaux. D'un point de vue de l'exploitation, seulement deux niveaux d'abstraction peuvent être considérés :

- un niveau CR pour les techniciens de maintenance afin qu'ils puissent maintenir en état le système,
- un niveau fonctionnel, utile aux opérateurs de conduite du système afin qu'ils puissent déterminer si la mission peut être accomplie en fonction de l'état de santé actuel du système et ainsi prendre les décisions appropriées pour la sécurité éventuelle des biens, des personnes et de l'environnement.

Il appartient néanmoins aux intégrateurs d'implémenter eux-même leur système de supervision en fonction de leur propre découpage fonctionnel. En effet, le système, fournit par un sous-traitant, dispose de son propre découpage fonctionnel alors que chez l'integrateur, ce système peut correspondre au niveau CR. Les diagrammes de classes de la figure 1.1 définissent ces deux points de vue selon différents niveaux d'abstraction (sous-fonction, fonction, sous-système, système).

Ainsi, une résistance assurant une fonction au sein même d'une carte électronique peut n'avoir aucune existence dans le découpage de l'intégrateur puisqu'il va chercher à remplacer la carte électronique sur laquelle ce composant est soudé. Dans ce cas, le diagramme de classes de la figure 1.1a peut être considéré comme correct quel que soit le type de système envisagé et le producteur (intégrateur ou fournisseur) considéré. En effet, le niveau système chez l'intégrateur sera considéré comme un STCGD par le sous-traitant.

Deux niveaux de découpage sont alors considérés. Ils correspondent aux deux acteurs principaux agissant sur le système. Le premier, l'agent de pilotage, n'est concerné que par les pertes fonctionnelles de son système (perte de la fonction de radiocommunication, perte de la fonction de freinage...). Le second, l'agent de



(a) Point de vue concepteur (b) Point de vue exploitant

FIGURE 1.1: Deux points de vue de modélisation matérialisés par des diagrammes de classes

maintenance, doit ensuite remplacer les CR et réparer les composants non remplaçables du système afin de rétablir (dans le cas d'un dépannage) ou maintenir (dans le cas d'un remplacement stratégique) un fonctionnement nominal du système dans des normes de sécurité prédéfinies. L'objectif de la fonction de diagnostic est de traduire ces pertes fonctionnelles en termes de composants. Les systèmes de diagnostic sont souvent basés sur des modèles. Ils disposent de connaissances classées en superficielles, profondes et mixtes (Ariton, 2003). Depuis plus de 20 ans, une importante littérature en décrit de nombreux mécanismes issus du domaine de la FDI (Fault Detection and Isolation) ou de l'Intelligence Artificielle. La connaissance superficielle traduit la connaissance de l'expert en maintenance. Elle est basée sur l'expérience et l'observation. Elle associe aux symptômes les défaillances qui les ont provoqué. La connaissance superficielle est donc constituée d'une liste de symptômes, d'une liste de défaillances et de relations associant une

défaillance à un ensemble de symptômes. Les avantages de cette approche résident dans la mise à jour de cette connaissance qui est relativement facile et la mise en œuvre d'un raisonnement qui permet d'obtenir des résultats assez rapidement. Cependant, la nature expérimentale de la connaissance peut entraîner des difficultés à énumérer tous les symptômes et toutes les défaillances ainsi que leurs relations. L'expert interrogé peut être alors amené à des conclusions erronées, c'est-à-dire qu'il peut lui arriver d'associer une défaillance à des symptômes sans savoir qu'il s'agit d'un cas particulier et donc sans préciser les conditions dans lesquelles cette relation n'est plus vraie. Un autre inconvénient est que seules les défaillances répertoriées peuvent être diagnostiquées. Les relations de causes à effets sont un exemple de connaissance superficielle (Ariton *et al.*, 2008). La connaissance profonde est issue de la phase conception du système physique. Elle est basée sur un modèle de l'architecture et/ou du comportement du système physique. La description de l'architecture comprend le découpage du système en sous-systèmes, leurs caractéristiques et les différentes liaisons entre sous-systèmes. Les notions de composants et d'interconnexions sont alors introduites. La connaissance dite profonde concerne le comportement normal du système physique. Dans ce cas, il s'agit de la description du fonctionnement correct de chaque composant établissant les relations entre entrées et sorties, suivant les divers modes d'utilisation. Comme le système de diagnostic peut mettre en œuvre des liens causaux entre l'état des composants et les symptômes observés, il peut expliquer que le comportement du système est causé par ses propriétés structurelles. Le terme de modèle causal est aussi utilisé. La connaissance du comportement du système physique en état de panne est également de nature profonde. Elle associe à une défaillance l'ensemble des symptômes qu'elle provoque. Le concepteur peut aussi modéliser les réactions du système dans différentes situations de défaillance. La connaissance profonde regroupe donc la connaissance de la structure et du comportement normal voire de comportements anormaux. La connaissance profonde est disponible à l'issue de la phase de conception. Cela permet donc de disposer de connaissances en vue de la conception du système de diagnostic avant même l'exploitation du système. L'utilisation de modèles de comportement normal permet de détecter des fautes ou des dysfonctionnements qui n'ont pas été observés auparavant. Une partie des connaissances profondes peut être réutilisée pour le diagnostic d'autres systèmes qui possèdent des composants de même type. Ainsi, la mise à jour du modèle associé pourra être facilement réalisée si l'architecture du système physique est légèrement modifiée par le concepteur. Les systèmes de diagnostic utilisant aussi des modèles de comportement pour différents états de défaillance facilitent la mise

à jour de la base grâce à l'ajout de relations entre symptômes et défaillances. L'approche consistant à décrire le comportement normal du système physique paraît a priori intéressante. Cependant, elle comporte un inconvénient susceptible de remettre parfois en question l'efficacité du système de diagnostic dans certain cas. En effet, la modélisation détaillée de l'architecture et du fonctionnement normal des équipements est rendue difficile par leur complexité et leur diversité. Or, une modélisation trop générale ne permet pas de déduire des résulats intéressants ; c'est-à-dire désignant de façon précise les composants défaillants. Pour les systèmes de diagnostic mettant en œuvre des connaissances décrivant des comportements en état de défaillance, un inconvénient est dû à l'expression de la connaissance à partir de relations entre défaillances et symptômes car seules les défaillances répertoriées peuvent être diagnostiquées (Siontorou *et al.*, 2010). Certains systèmes de diagnostic destinés à des STCGD utilisent à la fois de la connaissance profonde et de la connaissance superficielle afin de bénéficier des avantages offerts par l'utilisation de chaque type de connaissances et de remédier le plus possible à leurs inconvénients. Souvent la connaissance superficielle est utilisée, dans un premier temps, pour diagnostiquer rapidement les cas de défaillances répertoriés, généralement les plus simples et les plus fréquents. L'utilisation de la connaissance profonde, dans un second temps, permet soit d'infirmer ou de confirmer les résultats alors obtenus, soit de compléter ou de diagnostiquer des cas de défaillances non résolus par l'exploitation de la connaissance superficielle. Cependant, les connaissances nécessaires dépendent du type de raisonnement. Malheureusement, l'accès à toute la connaissance n'est pas possible pour des raisons économiques. Il faut donc proposer une architecture permettant de surveiller des systèmes donnant un grand nombre d'informations et pour laquelle les connaissances de conception ne sont pas toujours accessibles.

L'objectif étant pour les constructeurs de STCGD, de proposer aux exploitants un système de diagnostic disponible dès les premières utilisations de tous nouveaux systèmes, pour lesquels il est impossible d'employer une connaissance expérimentale. La mise en œuvre de connaissances profondes dans le système de diagnostic s'impose par conséquent.

La modélisation du comportement normal consiste à décrire le fonctionnement précis de chaque système et sous-système. Or, il est très difficile de modéliser le fonctionnement détaillé de STCGD et la façon dont les différents composants des sous-systèmes et les sous-systèmes interagissent entre eux. Le choix de la modélisation du fonctionnement en état de panne est justifié par le fait que la connaissance est disponible. Les experts qui détiennent ces informations existent (ingénieurs de

conception) et une partie de la connaissance qu'ils ont accumulé est régulièrement consignée dans des documents tels que le TSM (Trouble Shooting Manuel). Cette approche nécessite également la modélisation du STCGD, c'est-à-dire des différents sous-systèmes et de leurs interactions. Dans la plupart des systèmes actuels, certaines fonctions doivent impérativement rester disponibles pour des raisons de sécurité. Pour cela les fabricants mettent en place une redondance d'équipements garantissant la sécurité des personnes. Cette redondance (active ou passive) permet de conserver un fonctionnement nominal du système malgré une défaillance d'équipement. En effet, si une redondance d'équipements existe, et si un équipement devient défaillant, un autre équipement prend le relais permettant ainsi une poursuite de la mission. Des équipements peuvent ainsi être implantés en au moins deux exemplaires dans un système... Lorsqu'un équipement devient défaillant, la fonction de redondance n'est plus assurée mais le système peut poursuivre sa mission car la fonction assurée par les equipements redondants n'est pas défaillante. La redondance peut être assurée de façon matérielle ou analytique (souvent assurée par du logiciel). Concernant, la redondance analytique, la limite vient de la capacité disponible de traitement de l'information (espace mémoire, temps de calcul). La redondance matérielle n'est possible que sous certaines limites de poids et d'espace disponible. Lorsqu'une redondance n'est pas possible, le recours à une reconfiguration du système peut être opéré. C'est-à-dire que l'objectif de la mission en cours peut être révisé en fonction de la baisse de performance de l'équipement défectueux. Ceci peut être illustré par le cas suivant : un avion biréacteur a un moteur qui devient défaillant. Il ne peut alors plus voler aussi haut et aussi vite, et, pour des raisons de sécurité des personnes, devra chercher à se poser au plus près de l'endroit où il se trouve. Si, un modèle de vol existe pour ce cas, le logiciel de commande de vol peut compenser les effets de cette défaillance et ainsi faciliter son pilotage et/ou interdire certaines manœuvres. Lors d'une reconfiguration, il peut arriver que des équipements se mettent à communiquer entre eux, alors qu'ils ne le faisaient pas avant, pour pallier, isoler ou limiter les effets du composant défaillant.

1.2.4. Architecture de contrôle et de surveillance

Ces vingt dernières années, les architectures d'automatisme ont subie une forte évolution principalement due à l'émergence des Nouvelles Technologies de l'Information et de la Communication (NTIC). Ces changements successifs sont liés, d'une part, à l'évolution des besoins des utilisateurs et, d'autre part, au développement des technologies. Nombre d'architectures actuelles mettent en œuvre

des bus standard de communication (Ethernet, I2C, Modbus, CAN) comme le présente Cena *et al.* (1995). Jusqu'aux années 1980, les automatismes utilisant principalement des automates programmables industriels (API) ne supportaient essentiellement que des traitements séquentiels. Les tâches des API consistaient principalement à :

- gérer des demandes d'exécution en fonction de l'état de l'automatisme (image des entrées),
- élaborer des demandes d'exécution d'actions (positionnement des sorties).

Par la suite, les API ont géré de nombreuses fonctions complémentaires comme des fonctions métier, des fonctions de diagnostic du système et de l'application... Les automatismes centralisés géraient tout un ensemble de fonctions n'interagissant pas forcément entre elles. Lorsque l'usine ou un système automatisé était doté d'un API, les automaticiens qui devaient intégrer une fonction supplémentaire se posaient la question sur la capacité de l'automate ou du système d'automatisme en place à gérer les entrées et sorties supplémentaires et de la place mémoire disponible pour implanter cette fonction. Bien souvent, l'ajout d'une fonction supplémentaire pouvait être implantée sur l'API existant, même si elle n'avait aucun rapport avec l'automatisme résident (Bolton, 2010). Ces automatismes centralisés amenaient des nombreuses contraintes comme :

- l'absence d'autonomie des différents sous-ensembles,
- la difficulté de mise en service et de maintenance du fait des quantités d'entrées et de sorties gérées et la complexité des logiciels,
- l'absence de toute automatisation en cas de défaillance de l'API ou d'arrêt pour sa maintenance.

Ces contraintes induites par les systèmes d'automatisation centralisés ont conduit les concepteurs d'automatismes à envisager une segmentation de l'architecture. Cette segmentation est le plus souvent réalisée selon les entités fonctionnelles. Elle permet de simplifier l'automatisation par l'utilisation de plusieurs unités de traitement de l'information ou automates dotés chacun d'un nombre restreint d'entrées et de sorties et effectuant des traitements moins complexes. Ceci contribue à faciliter la mise en service et la maintenance du système automatisé. Cette segmentation a généré le besoin de communication entre unités de traitement et/ou automates. La communication est devenue un élément clé du bon fonctionnement des architectures d'automatismes répartis. Des réseaux locaux industriels (RLI) ont donc été développés afin d'assurer une communication efficace entre les différents API et unités de traitement de l'information répondant aux contraintes de sécurité, de fiabilité et de transmission des informations dans le respect des contraintes

temporelles. Un avantage induit par cette forme d'automatisation est la réduction du coût de câblage par la proximité accrue des unités de traitement et API des éléments qu'ils surveillent et/ou contrôlent. Cependant, sur des sites ou des équipements plus étendus, il est souvent nécessaire de gérer un nombre de points diffus importants et de prendre en compte les fonctions métiers réparties (variation de vitesse, dialogue homme/machine, pesage...). La réponse des constructeurs de produits d'automatismes est arrivée avec les réseaux et bus de terrain. Ceux-ci ont permis de gérer des entrées et sorties décentralisées puis de rendre accessibles des services (diagnostic, programmation) (CIAME, 2009).

De nouveaux langages de programmation et des mécanismes d'échange de données dans le monde informatique sont apparus comme le réseau Ethernet, et le protocole de communication TCP/IP. Ils ont commencé à être déployés sur des architectures d'automatismes. La généralisation de ce réseau et de ce protocole autorise une prise en compte simple de nouveaux outils de conduite et d'exploitation des données : MES (Manufacturing Execution System) et ERP (Enterprise Ressource Planning). Parallèlement à l'apparition du MES, les postes de supervision ont tendance à se repositionner au niveau de la conduite. Dans la terminologie NTIC, il ne faut pas comprendre seulement l'utilisation d'Ethernet et TCP/IP mais également les technologies et mécanismes éprouvés et utilisés dans le monde informatique tels que :

- les langages JAVA, HTML, XML,
- des nouvelles technologies issues du monde des télécommunications telles que WAP, Bluetooth, WML, WiFi, WiMax...
- des intergiciels (middleware) prenant en charge nombre de fonctionnalités nécessaires à la communication entre éléments de l'automatisme comme CORBA.

Un middleware (ou intergiciel) est une plateforme réunissant un ensemble de services et de fonctions réutilisables et extensibles indispensables au bon fonctionnement des applications dans un environnement réseau (Puder *et al.*, 2005). La tâche principale de cette plate-forme de services est de fournir un support d'interconnexion entre les éléments distribués. Les NTIC permettent ainsi :

- de repenser l'approche de la conception des architectures d'automatisme (Cavalieri *et al.*, 1995),
- d'augmenter les capacités et les performances des réseaux de communication par le développement de nouveaux modèles de communication (Schantz et Schmidt, 2002),

– de développer des services associés aux métiers des différents acteurs du système automatisé (Stipanicev et Marasovic, 2003).

Les systèmes sont donc mis en œuvre de façon très variés. Ils font appel à des données de natures différentes propres aux fonctions et aux technologies implémentées. Afin de les surveiller et de les contrôler, il faut assurer une interface entre le système et l'outil d'aide à la décision de maintenance. La figure 1.2 montre la structure à mettre en œuvre pour assurer la surveillance des composants. La communication de ces données a pour finalité la génération de symptômes. Les composants ayant une certaine hétérogénéité, il faut choisir des algorithmes appropriés aux technologies.

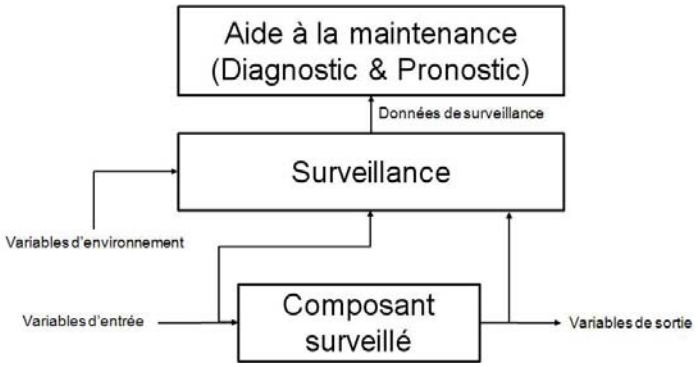


FIGURE 1.2: Principe de surveillance d'un composant pour l'aide à la maintenance.

1.3. Aide à la maintenance conditionnelle et à la planification des systèmes

1.3.1. Problématique de maintenance conditionnelle et de l'évaluation de l'état de santé

L'objectif de la CBM (Condition Based-Maintenance), ou maintenance préventive conditionnelle, est d'identifier les prémices de fautes ou de défaillances avant quelles ne deviennent critiques. Ceci permet une planification plus précise de la maintenance préventive (Bengtsson *et al.*, 2004). Elle est une alternative aux politiques de maintenance corrective et préventive systématique (Garga *et al.*, 2001). La CBM est basée sur des analyses des tendances de grandeurs caractéristiques de la santé du système ou de ses composants par rapport à des limites prédéterminées dans le but de détecter, analyser et corriger les problèmes de l'équipement bien avant que les défaillances ne se manifestent (Azam *et al.*, 2002). En maintenance

préventive systématique, l'opération intervient après un certain historique d'utilisation ou suivant un intervalle de temps basé sur des probabilités de défaillance. Ce type de maintenance est propice à la mise au rebut de composants qui aurait pu continuer à rendre leur service encore quelque temps. En ce qui concerne la maintenance corrective, elle intervient après l'apparition de la panne ou lorsque celle-ci dépasse les limites de fonctionnement spécifiées, ce qui signifie donc que la pièce va être utilisée sur toute sa durée de vie, présentant un avantage par rapport à la maintenance planifiée. Cependant, le temps de préparation de l'équipe de maintenance, le temps de recherche du ou des composants défectueux et la possibilité que le ou les composants de rechange soient indisponibles, font que ce type de maintenance peut être coûteux et peut immobiliser longuement le système subissant l'opération de maintenance. La mise en œuvre de la CBM requiert une instrumentation spécifique du système ou de certains de ses composants par un ou plusieurs capteurs dont les mesures seront traitées afin d'extraire les grandeurs représentatives de l'état de santé du système ou de ses composants qui, eux-mêmes, seront aussi traités soit pour définir les composants défaillants dans un but de diagnostic ou encore pour estimer les temps de vie résiduels constituant ainsi une forme de pronostic quant à la possibilité d'engagement du système. De cette façon, la mise en œuvre de la CBM concourre, par les traitements qui lui sont nécessaires, à la recherche des composants défectueux raccourcissant une intervention de maintenance corrective et allongeant l'utilisation des composants. Cela conduit à une réduction du nombre d'interventions de maintenance corrective et une meilleure planification. Les traitements nécessaires à la CBM produisent aussi des données pour l'aide à la décision concernant les actions de maintenance sur les équipements ou leur engagement. Les traitements de la CBM sont effectués en trois étapes : l'acquisition de données, le traitement des données et la prise de décision de maintenance. Le pronostic et le diagnostic sont deux aspects très importants faisant parties intégrantes de la CBM (Jardine *et al.*, 2006).

La mise en place d'une fonction pronostic requiert l'utilisation de l'état de santé courant, d'un historique des états de santé, d'un historique des opérations de maintenance passées et de l'usage futur prévu de l'équipement (prévision de la charge et du temps de charge de l'équipement dans un certain intervalle de temps). Le pronostic étant une prédiction, l'incertitude doit être prise en compte dans son établissement.

Le but du projet OSA-CBM (Open System Architecture – Condition Based Maintenance) (Provan, 2003) a été de définir une architecture pour le diagnostic et le pronostic des systèmes ainsi qu'une méthode de travail afin d'implémenter les

traitements destinés à la CBM selon une architecture ouverte mettant en œuvre des ORB (Object Request Broker) afin d'assurer la communication. Dans OSA-CBM, sept couches ont été définies, permettant de mettre en place la maintenance conditionnelle d'un système :

1. Acquisition de données (Data Acquisition layer) est la couche d'acquisition qui vise à transformer un stimulus en signal électrique. Cette couche redéfinie en fait ce qu'est un capteur.

2. Manipulation de données (Data Manipulation layer) est la couche qui traite le signal de la couche d'acquisition afin d'être envoyé et utilisé par le système pour en extraire les grandeurs représentatives de l'état de santé du système ou de ses composants.

3. Surveillance de l'évolution des données (Condition Monitoring/State Detection layer) est la couche qui extrait les données des couches d'acquisition de données et de manipulation de données afin de les comparer à des valeurs limites prédéfinies. Lorsque les limites sont franchies, cette couche génère des alarmes et des symptômes.

4. Diagnostic (Health Assessment layer) est la couche qui reçoit les données de la couche de surveillance et d'autres modules de diagnostic pour déterminer si la santé du système, sous-système ou composant surveillé est dégradée. Si sa santé est dégradée, le module de diagnostic génère un diagnostic sur une ou plusieurs conditions de fautes associées à un niveau de confiance. Le diagnostic doit prendre en compte l'évolution de la tendance à partir de l'historique de santé, de la charge, du statut opérationnel et de l'historique de maintenance.

5. Pronostic (Prognostics layer) est la couche qui, en fonction de l'approche mise en œuvre, peut recevoir des données de toutes les couches précédentes. Son principal objectif est de projeter l'état de santé actuel du système et de ses composants dans le futur en prenant en compte les futurs engagements que l'équipement devra assurer. Un état de santé est alors fourni comportant des informations sur les temps de vie résiduels du système et de ses composants.

6. Aide à la décision (Decision Support layer) est la couche qui fournie des recommandations et des alternatives pour le maintient en bon état du système. Elle exploite pour cela les données des couches de diagnostic et de pronostic. Les recommandations peuvent être des actions de maintenance planifiées, la modification de la configuration opérationnelle du système afin d'atteindre l'objectif du futur engagement ou bien de revoir les objectifs du futur engagement ou d'affecter à l'équipement un autre futur engagement.

7. Présentation (Presentation/GUI layer) est la couche qui assure l'interface entre le système de CBM de l'équipement et un ou plusieurs opérateurs humains, elle sert notamment à présenter les informations élaborées par le système de CBM.

Dans la suite de ce mémoire, les travaux présentés contribuent aux couches de diagnostic et de pronostic (Health assessment and prognostics layers). Une difficulté de la mise en place d'outils de la CBM réside dans la capacité à diagnostiquer et à pronostiquer le système étudié. Il s'agit de répondre notamment aux questions suivantes. Quelles méthodes de diagnostic et de pronostic choisir ? Avons-nous la connaissance suffisante afin d'assurer un diagnostic et un pronostic fiable du système ? L'utilisation des résultats de pronostic corrélés aux charges opérationnelles des futures missions du système permettent de définir si l'objectif peut être atteint. Il s'agit aussi d'un objectif de la PHM (Prognostic and Health Management). Un système de PHM doit être capable de gérer les anomalies, les données de diagnostic et de pronostic au niveau des composants du système et de pouvoir remonter ces informations au niveau des fonctions, des sous-systèmes et du système global. Roemer *et al.* (2007) indiquent que les algorithmes de diagnostic et de pronostic doivent être mis en place au plus près du composant et remonter ensuite la décomposition hiérarchique du système. La PHM réside principalement sur la mise en place efficace d'une fonction de pronostic. Des études ont déjà été menées portant principalement sur les machines tournantes, comme des réacteurs d'avions (Wilkinson *et al.*, 2004) ou bien encore les boites de vitesses (Lebold *et al.*, 2000; Garcia *et al.*, 2006). Le diagnostic et le pronostic sont des processus d'évaluation de l'état de santé d'un système. Le diagnostic est une évaluation de l'état de santé courant et passé du système basée sur des symptômes observés. Le pronostic est une évaluation de l'état de santé futur (Mathur *et al.*, 2001). Les fonctions de diagnostic et de pronostic apparaissent donc comme étant complémentaires pour la détermination de l'état de santé d'un système.

1.3.2. Types de diagnostic

Dans la première partie de cette section, nous donnons des définitions des concepts utiles à la compréhension de la problématique du diagnostic. Dans la littérature associée à ce domaine, plusieurs définitions parfois divergentes apparaissent. Les définitions proposées nous permettent d'exprimer notre point de vue sur la surveillance et le diagnostic de systèmes techniques.

Dans les industries manufacturières et de process, de nombreux développements ont été menés pour élaborer des produits de qualité, réduire les rebuts, et satisfaire les normes environnementales et de sécurité. Pour répondre aux nombreuses

exigences (normes, marchés...), les processus industriels modernes possèdent un nombre important de variables de surveillance contrôlées le plus souvent en boucle fermée. Les systèmes de contrôle sont conçus de sorte à présenter une certaine robustesse vis-à-vis des perturbations possibles envisagées et des changements qui apparaissent dans le processus. Bien que ces contrôleurs puissent compenser beaucoup de ces perturbations, il en existe, souvent non-envisagées lors de la conception, que les contrôleurs ne peuvent pas compenser. Elles conduisent alors le système au-delà des limites figurant dans ses spécifications. Ces changements sont appelés des fautes.

Une faute est un écart non permis d'au moins une propriété ou d'une variable caractéristique d'un système par rapport à son état normal, spécifié (Isermann, 1997; Chiang *et al.*, 2001). Les fautes sont des écarts du comportement normal d'un processus ou de son instrumentation. Les fautes peuvent être classées selon une échelle de sévérité allant de la détérioration de performance au dysfonctionnement partiel et parfois jusqu'à la panne totale. Elles apparaissent dans les équipements technologiques, les instruments de mesure (capteurs), les moyens de commande (actionneurs). Lorsqu'une faute dépasse un certain seuil de tolérance, elle provoque la défaillance du ou des composants.

La défaillance est une altération ou une cessation de l'aptitude d'un ensemble à accomplir sa ou ses fonctions requises avec les performances définies dans les spécifications techniques. L'ensemble est indisponible suite à la défaillance (Zwingelstein, 1995). Les défaillances peuvent avoir une dynamique lente comme les phénomènes d'usure ou très rapide voire soudaine comme cela est le cas pour la rupture d'un câble. Il est alors question de défaillance cataleptique lorsque celle-ci entraîne la défaillance totale du système.

Une défaillance résulte toujours de l'occurrence d'une faute qui, si cela est possible technologiquement, peut être révélée sous la forme d'un ou plusieurs symptômes par une fonction de détection et c'est ce qui permet de la caractériser. Un symptôme est une preuve sensorielle et observable d'une défaillance. C'est-à-dire un changement d'une quantité observable du comportement normal spécifié (Isermann, 1997).

Lorsqu'un système est défaillant, différentes étapes associées à des techniques permettent d'identifier la cause de la défaillance.

- La première étape consiste en une phase de détection de faute. La détection de faute a pour but de déterminer s'il y a présence de fautes dans le système et les instants auxquels elles sont apparues (Isermann, 1997). Cette phase

indique les changements de comportement et les fautes des composants du système (capteurs, actionneurs, etc.) (Patton *et al.*, 1989).

- La deuxième étape est une phase d'isolation de faute dont le but est de localiser et de déterminer le type de faute (Isermann, 1997). Pour Patton (1997), l'isolation de faute permet, après la détection de fautes, de déterminer l'équipement ou le composant défaillant à l'origine de la faute. Ce composant peut être un élément élaboré comme un capteur, un actionneur, une alimentation en énergie, un calculateur, un convertisseur de puissance ou un élément simple un câble, un joint, un pallier, un arbre de transmission...
- La troisième étape est une phase d'identification de faute dont le but est de déterminer l'amplitude et le comportement dans le temps de la faute (Isermann, 1997).

Ces trois étapes constituent les fonctions de surveillance dont l'objectif est d'élaborer et de mettre à disposition des informations structurées sur la situation du système observé. Ces fonctions de surveillance mettent en œuvre des mécanismes d'observation, de détection et de filtrage pour générer des indicateurs pertinents.

Actuellement, la surveillance de l'ensemble des dégradations potentielles d'un équipement est irréaliste pour deux raisons majeures :

- l'instrumentation disponible sur les systèmes est insuffisante et elle serait trop coûteuse à compléter,
- certaines dégradations sont techniquement inobservables les rendant impossibles à diagnostiquer.

Afin de pallier ce problème, des études sont menées et ont débouché sur la définition d'une propriété d'un système : la diagnosticabilité. Elle correspond à la capacité d'un système et de ces fonctions de surveillance et de diagnostic à exhiber des observations différentes pour chaque situation de faute anticipée (Ribot, 2009). Ce travail doit être entrepris durant la phase de conception du système et doit permettre de définir des indicateurs uniques pour chaque cause de fautes connue afin d'assurer un résultat plus fiable de la fonction diagnostic.

Pour développer la surveillance et faciliter la maintenance du système, une fonction de diagnostic, qui exploite les observations de la surveillance peut être mise en œuvre. Le diagnostic localise puis identifie la faute en réponse à une alarme. Il est effectué à partir du modèle de représentation, résultant de la surveillance. Il permet de mettre en évidence les fautes. Les dysfonctionnements sont localisés puis identifiés. Les règles de métier peuvent permettre d'identifier les causes des fautes. Le diagnostic établit la liste des éléments incriminés. Il doit rendre compte

des anomalies identifiées mais également des fautes dont la cause est indéterminée mais pour lesquelles un traitement est souhaitable (Combacau *et al.*, 2000).

En général, le diagnostic met en œuvre les trois étapes de la surveillance. Les approches de diagnostic développées actuellement par la communauté de l'automatique sont focalisées essentiellement sur les techniques de génération et de structuration des résidus ou sur des représentations multi-modèles (Niemann *et al.*, 2007).

Par ailleurs, la supervision a pour objectif d'assurer la gestion réactive et sûre des modes de fonctionnement d'un processus. Ces modes sont définis à partir de l'ordonnancement des tâches prévisionnelles, de la connaissance du système et des savoir-faire des opérateurs. La gestion réactive et sûre met en œuvre des méthodes de surveillance, de décision, d'accommodation de la commande aux fautes (Combacau *et al.*, 2000).

Les principales méthodes utilisées en surveillance et diagnostic sont synthétisées sur la figure 1.3.

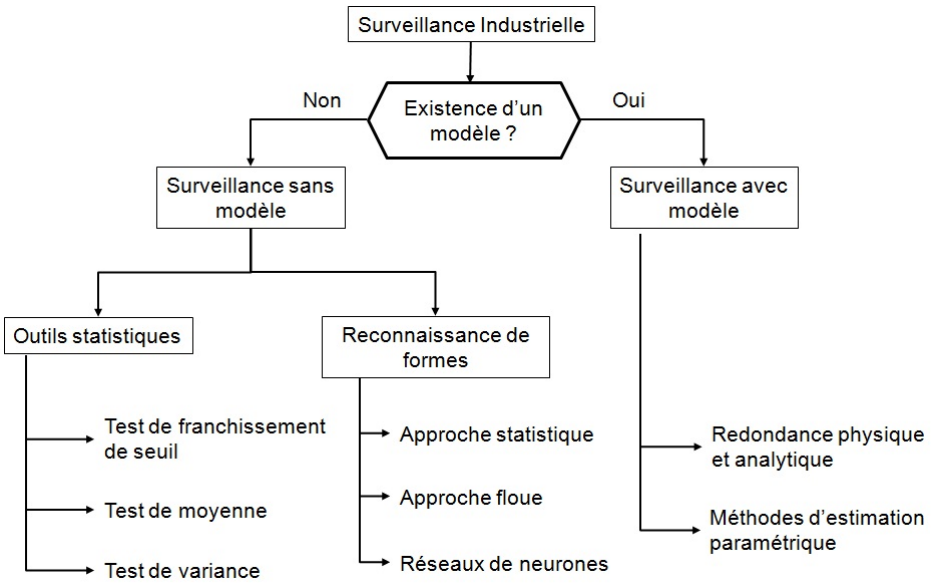


FIGURE 1.3: Classification des méthodologies de surveillance industrielles d'après (Zemouri, 2003)

Zemouri (2003) et Chiang *et al.* (2001) exposent différentes méthodes de diagnostic, basées sur des modèles, basées sur des données (Principal Components Analysis, Fisher Discriminant Analysis, Partial Least Squares, Canonical Variate Analysis), analytiques (observateurs, relations de parité) ou basées sur la connaissance (analyse causale, systèmes experts, reconnaissance de formes, systèmes hy-

brides). De nombreux travaux ont été effectués dans le domaine de la détection et de l'isolation de fautes (FDI) (Isermann, 1997; Glavaski et Elgersma, 2001; Halder *et al.*, 2004) et les outils développés sont souvent issus du domaine fiabiliste (Papadopoulos, 2003; Wu et Campion, 2004) ou encore de l'Intelligence Artificielle (IA) (Liu, 1996; Chen et Lee, 2002).

Durant son utilisation, le système peut être sujet à diverses défaillances impactant son mode de fonctionnement. Lors de l'apparition d'une défaillance, si celle-ci entraîne la défaillance soudaine sans signe avant coureur du système, la défaillance est qualifiée de cataleptique.Une défaillance peut être simple, multiple ou cachée (Dievart *et al.*, 2009). Une défaillance dite de dysfonctionnement contraint le système à opérer en mode dégradé impliquant donc une baisse de performance. La dégradation de performance peut affecter localement une fonction ou globalement le système. Lorsque la défaillance est dite fugitive, le système rentre dans un mode intermittent. C'est-à-dire qu'il alterne état normal de fonctionnement et état anormal. Ce type de défaillance peut être non signalé ou non expliqué. Cette typologie des défaillances est détaillée sur la figure 1.4.

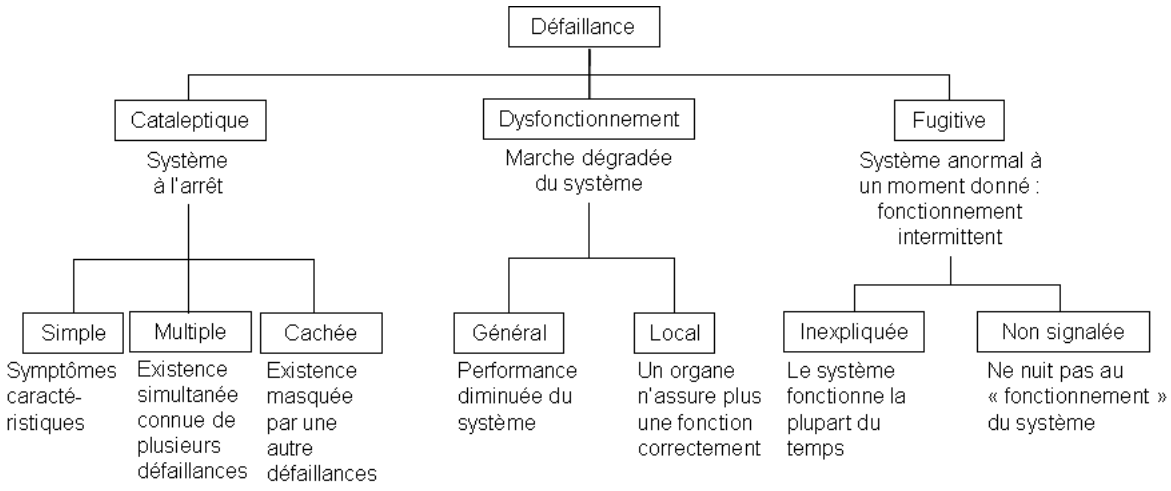


FIGURE 1.4: Typologie des défaillances

Il est nécessaire d'avoir une certaine connaissance du système pour lequel on souhaite mettre en œuvre des traitements de diagnostic. Différentes méthodes, desquelles la connaissance peut être extraite, peuvent être utilisées à l'élaboration d'un diagnostic. Ces différentes méthodes sont issues principalement d'études menées durant la phase de conception du système. A cela peut s'ajouter des outils de retour d'expérience qui permettent d'affiner la connaissance a priori du système

issue de la phase de conception. Des détails de ces méthodes sont donnés dans (Villemeur, 1988) et (Zwingelstein, 1995).

Il convient également de considérer les temps nécessaires à l'émission des observations (qui comprend le temps de détection, de traitement et de transmission) par le système de surveillance. Les traitements de diagnostic peuvent être effectués en ligne ou hors ligne. Le diagnostic hors ligne est basé sur toutes les observations (les symptômes) émises par le système de surveillance au cours de son exploitation. Dans certain cas, comme dans les systèmes critiques que sont certaines applications aéronautiques, le diagnostic hors-ligne est imposé par manque de connaissance et/ou de capacités de stockage. La complexité de certains systèmes ne permet pas non plus d'établir un diagnostic immédiat et l'opérateur de maintenance doit alors mettre en œuvre une procédure de remise en état (Troubleshooting Procedure) pour identifier les composants défaillants. Le diagnostic en ligne consiste quant à lui à diagnostiquer un système pendant son fonctionnement et donc pendant que celui-ci émet un flux d'observations. La tâche de diagnostic est alors effectuée lors de la réception d'une observation ou dans des fenêtres de temps pré-définies. Ceci suppose donc que la connaissance utile au diagnostic est disponible pour l'exécution de la fonction de diagnostic.

1.3.3. Pronostic et évaluation de l'état de santé

1.3.3.1. Santé des STCGD

La santé des STCGD est définie par les résultats de la fonction de diagnostic et les résultats de la fonction pronostic. Contrairement à la fonction diagnostic, la fonction pronostic permet de prévenir les défaillances en définissant le temps avant défaillance des composants. Le choix est laissé libre à l'utilisateur d'utiliser à bon escient ces données afin que le composant soit remplacé, réparé ou entretenu avant sa défaillance. L'utilisation des données de pronostic dans la fonction de diagnostic peut également s'avérer utile lors de la prise de décision concernant la déclaration ou non de l'état de défaillance d'un composant donné. L'incrimination par la fonction de diagnostic d'un composant associé à un temps de vie résiduel (Remaining Usefull Lifetime - RUL) relativement faible établie par la fonction pronostic constitue un bon apport pour l'aide à la décision.

1.3.3.2. Définitions du pronostic

Le terme pronostic provient du grec progignôskein qui signifie "connaître à l'avance". C'est une notion utilisée en médecine où elle concerne la prévision, après le diagnostic, du degré de gravité et de l'évolution ultérieure d'une pathologie, y

compris son issue, en se référant à l'évolution habituellement observée pour des troubles similaires chez de nombreux autres patients. Cette définition est transposable dans le cadre de la maintenance prévisionnelle de systèmes techniques. Ainsi, le rôle du pronostic consiste à prédire l'état futur d'un équipement à partir de son état actuel et passé (antécédents) et, éventuellement, en tenant compte de son usage futur.

Pour Mathur *et al.* (2001) et Engel *et al.* (2000), le pronostic est la capacité à fournir une détection et une isolation précoce d'un état de faute naissant vers un état de défaillance d'un composant ou d'un sous-ensemble, et d'avoir la technologie et les moyens de contrôler et prédire la progression de cet état de faute vers la défaillance d'un composant. Si cette définition explique ce qu'est le pronostic, elle ne précise pas quels sont les méthodes, techniques ou outils à mettre en œuvre afin d'établir un pronostic. Byington *et al.* (2002), définissent le pronostic comme la capacité à prévoir l'état futur d'une machine en se basant sur le diagnostic de l'état actuel de la machine et sur l'historique des défaillances et des opérations. Cette définition présente des éléments permettant d'établir un pronostic mais n'est pourtant pas suffisante car elle n'intègre pas la notion d'utilisation future de l'appareil comme dans (Lebold et Thurston, 2001) où le pronostic est défini comme la capacité à fournir une prévision fiable et suffisamment précise du temps de vie résiduel restant d'un équipement en service. La fonction principale du pronostic est de projeter dans le futur l'état de santé courant de l'équipement en prenant en compte l'utilisation future. Vachtsevanos et Wang (2001) expliquent que dans le secteur industriel et de la fabrication, le pronostic est interprété comme la durée de vie résiduelle restant à une machine ou un composant une fois un état imminent de défaillance détecté et identifié.

Le pronostic est ainsi la capacité à fournir une estimation fiable de l'état de santé futur d'un composant (ou d'un sous-ensemble) en se basant sur le diagnostic de son état actuel, l'historique des défaillances et des opérations de maintenance et l'utilisation futur du composant afin de donner une estimation de la durée de vie résiduelle du système.

Toutefois, d'un point de vue "Surveillance/Diagnostic" la signification du pronostic n'est pas tout à fait la même. Le pronostic est assimilé à un diagnostic prédictif, qui s'exerce sur la dégradation et non pas sur la défaillance comme le diagnostic classique. "L'objectif du pronostic (diagnostic prédictif) est d'identifier les causes et de localiser les organes qui ont entraîné une dégradation particulière" (Zemouri, 2003).

En fonction des besoins et des contraintes de maintenance, le pronostic correspond donc à :

- l'estimation du temps restant avant la défaillance et du risque d'apparition d'un ou de plusieurs autres modes de défaillance existants ou à venir. Cette estimation correspond donc à la durée de vie résiduelle (Remaining Useful Lifetime - RUL) d'un composant (Yan *et al.*, 2002; Luo *et al.*, 2003),
- la probabilité qu'une défaillance intervienne avant un instant donné. Le RUL peut être défini comme donnant le temps avant une probabilité de défaillance devenant inacceptable et mettant en danger l'intégrité des biens, des personnes et de l'environnement.

Le pronostic n'est pas une finalité. Il doit aboutir au déclenchement direct ou différé d'interventions de maintenance. Ceci nécessite, au préalable, de sélectionner la meilleure alternative de maintenance à partir du pronostic et de l'impact de chacune de ces alternatives sur les indicateurs de performances du système de production. Le problème décisionnel est complexe et nécessite le développement d'un processus d'aide à la décision complémentaire au processus de pronostic (Mathur *et al.*, 2001). Les méthodologies de pronostic peuvent être divisées en trois catégories :

- les approches exploitant un modèle physique,
- les approches guidées par les données,
- les approches basées sur l'expérience.

Le choix d'utilisation d'une de ces approches est fonction des données disponibles qui peuvent être fournies en entrée de la fonction de pronostic (Muller, 2005). Chacune d'entre elles est l'objet d'une description sommaire dans les parties suivantes. Elles sont cependant plus détaillées dans (Vachtsevanos *et al.*, 2006).

1.3.3.3. Pronostic basé sur les modèles

Le pronostic basé sur les modèles ("model-based prognostic" ou “physics of failure based prognostic" (Byington *et al.*, 2003)) fait appel à une fonction mathématique connue de la dynamique de la dégradation, surveillée par un indicateur, conduisant à la défaillance d'un composant. Le résultat de la fonction pronostic implémentant ce type d'approche est le temps de vie résiduel ou RUL du composant. La mise en œuvre est assez aisée si l'évolution dans le temps de l'indicateur associé à la dégradation est une fonction monotone. La dégradation est aussi souvent fonction du mode de fonctionnement des composants (Luo *et al.*, 2003). Ainsi, lorsqu'un changement de mode de fonctionnement intervient dans l'utilisation du système, la fonction de dégradation doit être mise à jour pour une dégradation donnée. Le problème de ce type d'approche vient du fait qu'il faut disposer des

indicateurs de surveillance de dégradation et de leur fonction d'évolution pour chaque mode de fonctionnement. Ceci n'est pas toujours possible. En effet, il apparaît que les mesures émanant d'un dispositif à surveiller ne sont pas forcément pertinentes pour la surveillance de l'état de dégradation du système. Il apparaît donc que l'implémentation d'une fonction de pronostic sur un système doit être établie dès la conception afin de pouvoir spécifier et lister les données nécessaires à la mise en œuvre de ce type de résultats. Des alternatives existent et nécessitent la construction d'un indicateur générique issu de différentes mesures qui mettent en œuvre des techniques de traitement du signal. Ces alternatives peuvent être utiles dans le cas où une surveillance des dégradations doit être établie sur un système déjà existant ne permettant pas d'évolution vers ce type de services ou ne pouvant pas accueillir de capteurs dont les mesures seraient utilisées à des fins de pronostic.

1.3.3.4. Pronostic guidé par les données

Le pronostic guidé par les données (Data-driven prognostic) est basé sur des méthodes statistiques (analyse de tendance, estimateur d'état...) afin d'exploiter les symptômes ou les indicateurs de la dégradation. La méthode consiste en une extrapolation de l'évolution des paramètres de surveillance. Il s'agit d'utiliser des modèles statistiques d'analyse de tendance ou alors d'exploiter les techniques mises en œuvre par le sous-processus de diagnostic à des fins de pronostic (Zwingelstein, 1995; Chiang *et al.*, 2001). Trois catégories de méthodes de pronostic guidé par les données sont recensées dans (Byington *et al.*, 2004a). Elles sont distinguées par la nature des techniques exploitées :

- le pronostic par analyse de tendance (Evolutionary/Feature-based prognostic") exploitant des modèles statistiques,
- le pronostic par apprentissage automatique (Machine learning /AI based prognostic) exploitation de modèles de type boîte noire issus de l'intelligence artificielle,
- le pronostic basé sur un estimateur d'état (State estimator prognostic).

1.3.3.5. Pronostic basé sur l'expérience

Lorsqu'il n'y a pas de modèle physique connu ou que celui-ci est trop complexe et qu'aucun dispositif de suivi de l'état de dégradation n'est opérationnel, le processus de pronostic peut être mis en œuvre par une approche basée sur l'expérience (Experience-based prognostic) (Byington *et al.*, 2002). Ce concept est basé sur la modélisation probabiliste de la dégradation, issue de l'expérience acquise sur le comportement du composant étudié.

La mise en œuvre de ce type d'approche est possible à condition de disposer

d'un volume de données issues d'un processus de retour d'expérience suffisant ou bien d'expertises. Ce type de pronostic est particulièrement bien adapté aux systèmes complexes impossibles à modéliser physiquement et/ou dont les indicateurs de dégradation sont trop sensibles aux fluctuations des conditions d'utilisation. Dans des conditions d'utilisation réelles, l'évolution de la fiabilité ou de la dégradation d'un composant est influencée par un ensemble de paramètres, l'environnement (température, humidité...) ou le mode de fonctionnement de l'équipement (charges de travail, régimes constants ou variables...). Ces deux méthodes de pronostic exploitent des données d'historiques dont seule la source diffère (enregistrement de données en continue ou retour d'expérience). Le résultat fourni à l'utilisateur ou à la fonction d'aide à la décision diffère aussi selon le type de connaissance exploitée. Le pronostic guidé par les données fournit le même résultat que le pronostic basé sur les modèles, à savoir un RUL alors que le pronostic basé sur l'expérience retournera un résultat issue du domaine fiabiliste, à savoir un MTBF (MeanTime Between Failure), un MTTF (Mean Time To Failure) ou une probabilité de défaillance sur un horizon temporel donné.

1.3.4. Aide à la maintenance

L'aide à la maintenance met en œuvre des méthodes pouvant être présentes dans la couche "d'aide à la décision" définie par le standard OSA-CBM. L'état de santé d'un équipement peut être défini par son diagnostic et son pronostic. Le résultat de ces fonctions constitue donc une aide à la décision quant aux opérations de maintenance à effectuer. Le pronostic, produisant un RUL peut être considéré comme une fonction complémentaire permettant d'affirmer, d'infirmer ou d'affiner le résultat de la fonction diagnostic. Si un composant est déclaré comme défaillant et qu'il possède un RUL faible, il pourra être considéré comme défaillant. S'il est accompagné d'un RUL assez élevé, cela veut dire que le composant est vraisemblablement défaillant et qu'il a été l'objet d'une défaillance cataleptique que la partie pronostic de la fonction de surveillance n'a pas été en mesure de détecter ou encore lorsque deux composants sont déclarés possiblement défaillants pour une même « panne » (la cause est l'un ou l'autre) l'opérateur de maintenance pourra choisir en priorité le remplacement ou la réparation de celui dont le RUL est le plus faible. Si le composant n'est pas défaillant, un RUL élevé n'impose pas de remplacement ou d'entretien du composant et un RUL faible signifie le remplacement ou l'entretien immédiat du composant. La détermination de seuils de décision pour lesquels la valeur du pronostic sera considérée comme faible aura toute son importance en termes de risques de défaillance lors du prochain engagement du

système. La décision devra aussi prendre en compte la vitesse d'évolution de la valeur et la confiance qui lui est associée. Un seuil défini trop haut reviendrait à faire de la maintenance préventive systèmatique avec un échéancier permettant une meilleure utilisation de la ressource mais sans être optimale. Un seuil défini trop bas et une gestion des erreurs, au niveau de la confiance dans la mesure et dans la charge théorique de la futur mission, reviendrait à subir la défaillance et donc à devoir mettre en place une maintenance corrective. Les responsables de la gestion de maintenance pourraient mettre en place des fonctions économiques permettant de définir la nécessité de remplacement ou d'entretien d'un composant en fonction du coût de l'intervention, de l'incertitude sur le RUL, des charges liées au futurs engagements de l'équipement, des coûts des rechanges ou des matériels de réparations et de leur détention.

1.4. Diagnostic décentralisé vs diagnostic distribué

1.4.1. Avantages et inconvénients des architectures

Différents types d'architectures peuvent être considérées lors de la conception d'un système. Cependant, chacune de ces architectures présente ses propres avantages et inconvénients. Ainsi, il existe différents types d'architectures de systèmes comme cela est présenté sur la figure 1.5 et commenté dans la suite de cette section (Zennir, 2004).

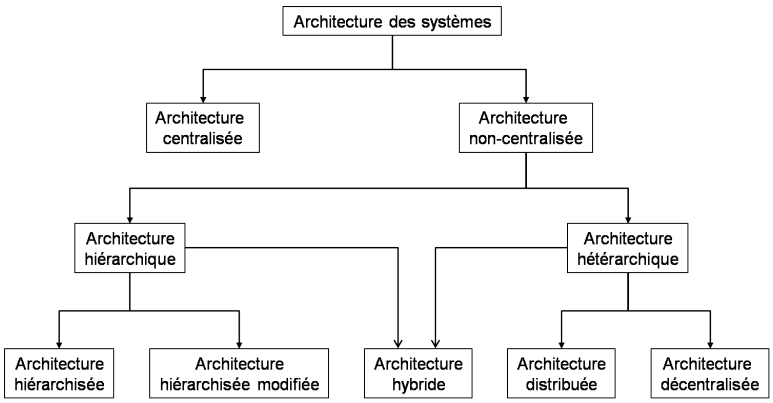


FIGURE 1.5: Différentes architectures de systèmes selon Zennir (2004).

Ces architectures organisent les agents de différentes manières. Un agent logiciel (computational agent) peut être défini comme une entité logicielle visant à résoudre

un problème, à effectuer une tâche. Cette entité est capable, au moins, de conduire la tâche pour laquelle elle a été conçue, d'interagir avec son environnement, de percevoir son environnement, de produire une réponse dans un temps donné et de prendre des initiatives quand cela est opportun (Jennings et Wooldridge, 1995).

Dans le cas d'une architecture centralisée, un seul agent planifie, prend les décisions, pilote tous les mécanismes de coordination et maintient l'information globale sur l'ensemble des autres agents. Ce type d'architecture est représenté sur la figure 1.6.

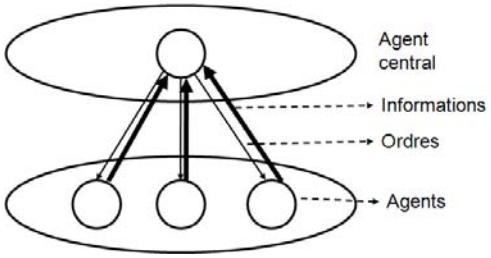


FIGURE 1.6: Architecture centralisée

Les avantages de cette structure sont les suivants :

- l'agent central dispose d'une vue globale sur le système complet,
- les communications entre agents sont réduites,
- le nombre d'unités de contrôle, de moyens de traitement et de gestion de l'information est limité,
- il est possible d'optimiser de façon globale la gestion de l'information.

Toutefois, cette architecture présente les défauts suivants :

- la vitesse de réponse dépend de la dimension du système (c'est-à-dire lorsque le nombre d'agents augmente la vitesse des communications décroît),
- le système est peu robuste car il est sensible aux fautes de l'agent central,
- il faut que l'agent central dispose des informations globales à chaque instant, ce qui n'est pas toujours faisable,
- il est difficile de faire évoluer le système à cause de la non modularité.

Des architectures non centralisées ont donc été développées pour pallier ces inconvénients. Deux types d'architectures non centralisées sont distingués. Les architectures hiérarchiques et les architectures hiérarchiques modifiées (Zennir, 2004).

Les architectures hiérachiques sont inspirées des structures sociales. Ces architectures sont composées de plusieurs niveaux de sous-systèmes avec, au niveau supérieur, une centralisation locale comme le montre la figure 1.7. C'est-à-dire

que chaque niveau dispose d'un agent central qui contrôle et coordonne les agents subalternes. Les relations entre les agents d'un même niveau sont indépendantes des niveaux supérieurs. Cependant, il y a une relation de maître (niveau supérieur) à esclave (niveau inférieur) entre les niveaux.

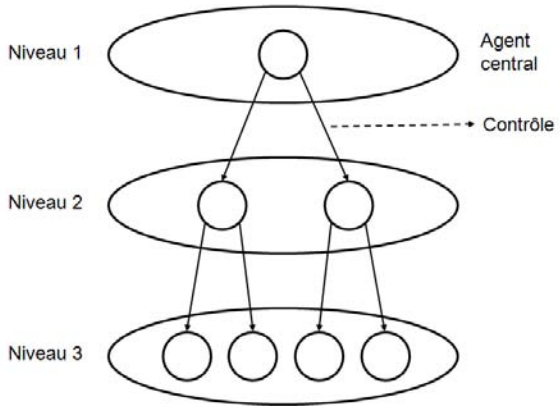


FIGURE 1.7: Architecture hiérarchisée

Il existe une autre forme d'architecture hiérarchique dans laquelle les agents d'un même niveau peuvent se coordonner entre eux et communiquer. Elle est appelée architecture hiérarchique modifiée. Parmi les avantages, de cette architecture décrite sur le shéma de la figure 1.8, les points suivants peuvent être relevés :

- une certaine conformité par rapport à la résolution classique des problèmes,
- des réponses plus rapides grâce au couplage maître/esclave entre les agents,
- une forme d'optimisation globale,
- la robustesse est plus importante que dans le cas d'une architecture centralisée, l'architecture est plus flexible par rapport au nombre d'agents et adaptative par rapport aux nouvelles situations des agents.

Les inconvénients sont dus :

- aux problèmes de transfert de partage des informations et de coordination entre les agents du même niveau,
- au problème d'évolutivité : pour effectuer des modifications de structure, il faut refondre tout le système et mettre à jour les structures de plus haut niveau dans l'architecture,
- au problème de perturbation imprévue, comme la panne d'une ressource qui invalide la planification et l'ordonnancement pour le contrôleur de niveau élevé,

– au problème de robustesse dans le cas où le contrôleur central de haut niveau tombe en panne qui correspond à une situation exigeant l'arrêt total du système.

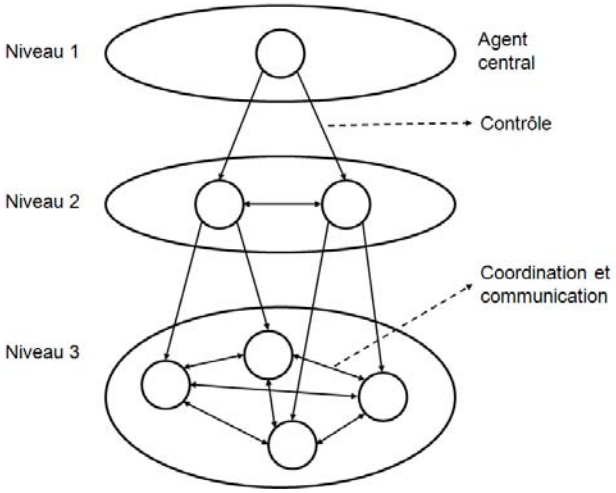


FIGURE 1.8: Architecture hiérarchique modifiée

Les architectures hétérarchiques sont une sorte d'opposition aux architectures hiérarchiques. En effet, il n'y a pas de relation maître/esclave. Les agents sont donc considérés au même niveau et ont tous les mêmes possibilités de participer à la prise de décision. Les architectures hétérarchiques sont une forme d'architecture décentralisée ou d'architecture distribuée. Un exemple d'architecture distribuée est représenté sur le schéma de la figure 1.9.

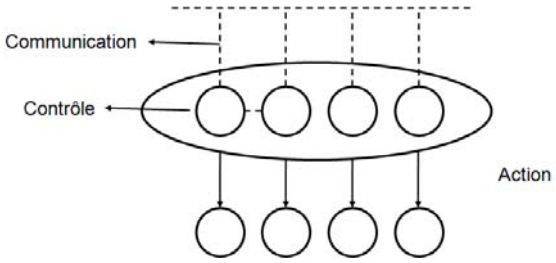


FIGURE 1.9: Architecture distribuée

Les agents de cette architecture disposent des propriétés suivantes :
– ils ont le même droit d'accès aux ressources,
– la communication entre eux peut être fortement développée,

- les agents sont indépendants pendant le fonctionnement,
- pour atteindre l'objectif du système global, la collaboration et le travail collectif sont nécessaires,
- les agents sont autonomes.

Parmi les avantages de cette architecture nous avons :

- l'amélioration de la flexibilité du système (possibilité d'ajouter ou de retirer facilement des agents),
- l'augmentation de la robustesse (tolérance aux fautes),
- l'amélioration de l'adaptation au changement des situations des agents ou de leur environnement,
- des agents autonomes et simples,
- le partage d'informations locales et globales.

Mais cette architecture présente aussi quelques inconvénients comme :

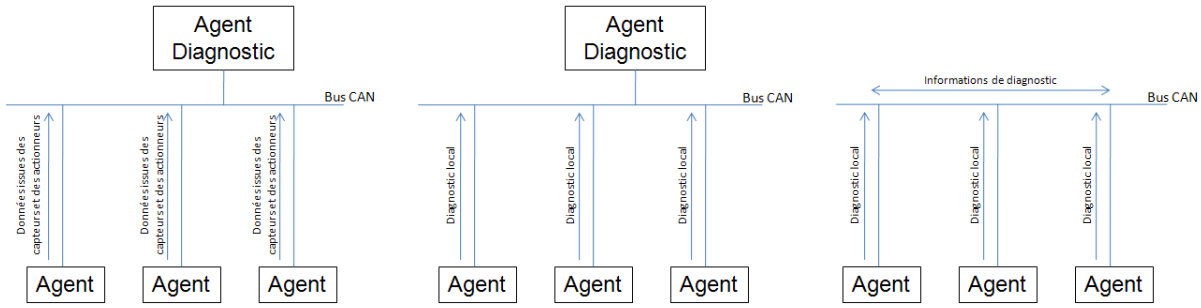
- la communication peut devenir très complexe,
- la coordination entre les agents est très importante et aussi complexe,
- l'atteinte de l'objectif global est incertaine, les agents cherchant tous à satisfaire leurs objectifs locaux,
- les performances globales du système dépendent du choix des règles locales et des protocoles de négociation entre les agents.

Parmi les architectures distribuées, il est possible aussi d'assimiler les architectures mettant en œuvre des web-services. Ainsi, Simoni (2007) définit un service comme une prestation immatérielle composable, manifestée de manière perceptible et qui dans une condition d'utilisation prédéfinie est source de valeur pour le consommateur et le fournisseur. Un système distribué est alors considéré comme un ensemble de domaines administratifs qui partagent leurs ressources afin de répondre aux besoins des utilisateurs locaux ou distants. Le traitement distribué nécessite la possibilité de décomposer le traitement en plusieurs tâches s'exécutant sur différentes machines reliées les unes aux autres. Les systèmes distribués résolvent donc d'abord le problème de la distribution des traitements. Trois types d'architectures distribuées peuvent être différenciés :

- les systèmes distribués fixes (hôtes et terminaux fixes) où la gestion de la mobilité n'est pas prise en charge,
- les systèmes distribués nomades (hôtes fixes et terminaux fixes ou mobiles) avec gestion de la mobilité personnelle
- les systèmes distribués ad hoc (hôtes et terminaux fixes et/ ou mobiles) qui mettent en œuvre des nouveaux mécanismes de la gestion de la mobilité.

Des travaux de développement de diagnostic embarqué ont été menés par Hallgren

et Skog (2005) et Biteus *et al.* (2008). Ces travaux ont conduit à l'implémentation d'un système de diagnostic distribué organisé autour d'un réseau embarqué dans des camions. Ce système est constitué d'agents locaux qui établissent le diagnostic du système dont ils ont la charge. Ces agents locaux coopèrent afin d'améliorer la fonction diagnostic. Les problèmes de conflits et de synchronisation du processus sont pris en compte. Ces développements ont été effectués à partir des travaux initiés par Reiter (1992). Ces auteurs ne considèrent que trois types d'architectures comme cela est indiqué sur la figure 1.10 : centralisées, décentralisées et distribuées. Un diagnostic centralisé, comme représenté sur la figure 1.10a, récupère toutes ces données issues des capteurs. Un unique agent établit le diagnostic. Le diagnostic décentralisé de la figure 1.10b a la même configuration à part que des diagnostics locaux sont envoyés au diagnostiqueur global. Le diagnostic distribué est basé sur le diagnostic décentralisé hormis le fait qu'il ne fait pas appel à un organe central de décision et de gestion des conflits (voir figure 1.10c)



(a) Diagnostic centralisé (b) Diagnostic décentralisé (c) Diagnostic distribué

FIGURE 1.10: Structures de diagnostic considérées dans Hallgren et Skog (2005); Biteus *et al.* (2008)

Dans le travail présenté dans ce mémoire, les termes architecture décentralisée et architecture distribuée sont confondus car, dans ces deux types d'architectures, il y a une distribution de l'intelligence et de l'autonomie. Les architectures hybrides complètent cette typologie. Elles combinent les architectures précédentes. Dans une architecture hybride, les agents de contrôle de même niveau hiérarchique sont interconnectés via un même moyen de contrôle. Ils sont capables de communiquer et de coopérer pour satisfaire leurs objectifs locaux. Lors de perturbations, l'ensemble des agents de contrôle peut demander de l'aide à leur agent de contrôle (de niveau supérieur) pour résoudre les problèmes détectés. Cette architecture est représentée sur le schéma de la figure 1.11.

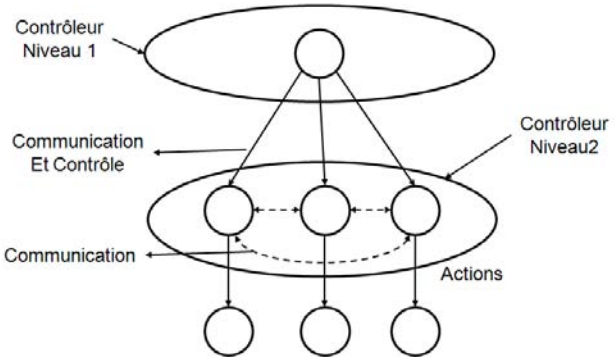


FIGURE 1.11: Architecture hybride

Les avantages des architectures hybrides sont ceux des architectures hiérarchiques et hétérarchiques. Cependant, des inconvénients subsistent comme la recherche du bon compromis entre la supervision au niveau hiérarchique et le degré d'autonomie attribué aux niveaux hétérarchiques. Ce compromis assure la stabilité et l'adaptation aux changements de situation d'un environnement complexe.

Il est possible de classer ces architectures en fonction de leur degré de décentralisation du plus faible au plus élevé : centralisée, hiérarchisée, coordonnée (équivalent à une architecture hiérarchique modifiée de Zennir (2004)), distribuée et distribuée supervisée (équivalent à l'architecture hydride de Zennir (2004)) et distribuée.

1.4.2. Diagnostic décentralisé

Plusieurs études ont été menées visant à déployer des fonctions de diagnostic de façon décentralisée afin de pouvoir fiabiliser la fonction de diagnostic et la rendre plus performante au niveau de la vitesse de convergence. Dans leur article, Console *et al.* (2007) proposent une structure de diagnostic basée sur les modèles décentralisés d'un système complexe représenté par des contraintes qualitatives. Les modèles sont distribués permettant la représentation des sous-systèmes. Des diagnostiqueurs locaux sont associés aux sous-systèmes et coordonnés par un superviseur. Ce dernier doit coordonner les diagnostiqueurs locaux et fournir des diagnostics globaux. L'approche établit un ensemble de diagnostics partiels qui représentent tous les diagnostics basés sur la cohérence. Différentes hypothèses sont faites comme celle expliquant qu'un diagnostiqueur local n'a pas la connaissance des autres diagnostiqueurs locaux. Ceci implique qu'il n'y a aucune coopération de bas niveau entre diagnostiqueurs. Il est possible aussi d'avoir une architecture hiérarchique des superviseurs correspondant à une décomposition hiérarchique du

système. La structure proposée utilise des modèles intemporels où les attributs dynamiques sont absents ou ont été retirés.

Dans les approches décentralisées de diagnostic, les travaux de Pencolé et Cordier (2005) qui portent sur les réseaux de télécommunications sont basés sur le fait que le modèle du système est de taille très importante, tandis que le diagnostic est lui-même relativement petit en termes de nombre de composants incriminés par rapport au nombre de composants implémentant le système. La raison de cette propriété est que les observations sur le système contraignent fortement les comportements possibles du système. Des diagnostics locaux sont établis puis fusionnés à l'aide d'un algorithme afin de définir l'état global du système supervisé. L'architecture et ces algorithmes permettent un diagnostic en ligne du système mais ne prend pas en compte l'ordre d'apparition des symptômes traités. A notre connaissance dans la littérature identifiée, la plupart des études sur le diagnostic décentralisé ont porté sur des systèmes à événements discrets (Boel et van Schuppen, 2002; Debouk *et al.*, 2003; Qiu et Kumar, 2006).

1.4.3. Apports des NTIC et des technologies distribuées au diagnostic

Les stratégies de maintenance et de diagnostic exploitent désormais les NTIC. En effet, des architectures sont développées mettant en œuvre des Systèmes Multi-Agents (Garcia-Beltran, 2004; Touaf, 2005) ou bien encore des Web-Services (Ardissono *et al.*, 2005; Sadovykh, 2005). Lorsqu'il est possible d'appeler le service de diagnostic depuis une connexion internet n'importe où dans le monde ce service devient du e-diagnostic. Li *et al.* (2005) distinguent quatre agents élémentaires afin d'assurer la maintenance conditionnelle d'un système : l'agent de surveillance, l'agent de diagnostic, l'agent de pronostic et l'agent de prise décision de maintenance. Il est possible de combiner ces agents afin de créer des modèles d'agents selon différents niveaux (machine simple, groupe de machines et entreprise). L'arrivée des NTIC a permis l'émergence d'architectures de type bus qui ont rapidement conduit à la définition de standards de connexion (Ethernet par exemple) et d'échanges d'informations (TCP/IP, UDP ...). Cette structure permet, grâce à la définition d'un protocole de communication, d'autoriser ou non des échanges entre agents connectés au bus. Ce protocole rend possible la création de comportements similaires à ceux rencontrés dans une architecture centralisée, hiérarchique, décentralisée...

1.4.4. Diagnostic distribué

Après avoir étudié les architectures décentralisées, des projets ont été lancés pour permettre l'implémentation d'architectures distribuées de diagnostic. En effet, elles permettent une implémentation sur un seul niveau hiérarchique de la fonction de diagnostic sans que l'un des agents ne doivent gérer les autres. Chaque agent de diagnostic est alors totalement autonome et a la possibilité d'initier une communication avec les autres agents de diagnostic quand cela est nécessaire.

L'architecture MAGIC est composée de différents agents permettant d'aboutir au diagnostic d'un système par sous-système ou global : Cette architecture comporte les agents suivants :

- le Data Acquisition Agent (DAA) qui est responsable de la gestion des données générées par le processus étudié à partir d'informations issues des capteurs et des actionneurs. Cet agent réalise aussi les pré-traitements des données (filtrage, formatage, validation, etc.). Il est également une interface entre le système étudié et le système MAGIC,
- les Diagnostic Agents (D-Agents) participent au processus de diagnostic global en effectuant des "tests de détection" et en produisant des symptômes qui seront examinés plus tard par l'agent de prise de décision diagnostic (Diagnostic Decision Agent - DDA). Plusieurs agents de détection peuvent fonctionner en parallèle. Ils sont déclenchés par le DDA suivant une stratégie de diagnostic définie a priori par l'ingénieur système,
- le Diagnostic Decision Agent (DDA) est le niveau de l'analyse de diagnostic qui conduit à une décision globale prenant en compte des résultats obtenus par les tests de détection transmis par les D-Agents,
- le Diagnostic Support Agent (DSA) apporte une assistance à l'opérateur pour les tâches de maintenance, où d'action de reprise de contrôle en cas de défaut,
- l'Operator Interface Agent (OIA) réalise l'interface Homme-Machine qui permet de visualiser les diagnostics fournis par le DDA ainsi que les différents symptômes et messages collectés sur le système. Il permet aussi d'interagir avec le système MAGIC en permettant notamment la désactivation de tests de détection. La figure 1.12 est une représentation sous forme de schéma des agents actifs en mode d'opération continue de l'architecture MAGIC.

Dans l'architecture MAGIC, le diagnostic est de type centralisé. En effet, le diagnostic fait référence à la supervision telle que nous l'avons définie. Dans cette

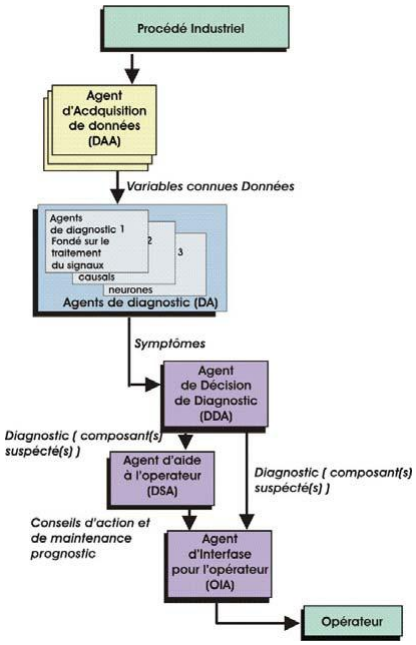


FIGURE 1.12: Agents actifs en mode d'opération continue de l'architecture MAGIC

architecture, la couche de surveillance est la seule qui soit véritablement distribuée. Les D-Agents, qui sont plutôt des agents de surveillance, implémentent des méthodes de générations de symptômes issues de domaines différents dont en voici une liste issue de Touaf (2005) :

- méthodes d'analyse des signaux (méthodes statistiques, méthodes fréquentielles, méthodes temps-fréquence),
- méthodes fondées sur la connaissance experte (modèles causaux),
- méthodes fondées sur des modèles analytiques (observateurs),
- méthodes fondées sur les modèles générés à partir des données (réseaux de neurones artificiels).

Chacun des D-Agents reçoit donc une mesure et génère un symptôme en fonction de la méthode qu'il implémente. Enfin, le DDA établit un état de santé global du système basé sur les symptômes générés. Cette architecture permet d'établir un diagnostic plus fiable basé sur des symptômes générés par des agents implémentant différentes méthodes mais exploitant les mêmes mesures. Ainsi, l'agent de diagnostic centralisé (le DDA) peut arbitrer ces diagnostics en fonction de la confiance qu'il accorde aux symptômes qui lui sont envoyés. Si un même symptôme est généré par tous les agents de surveillance, l'agent de diagnostic est sûr que la défaillance est présente. Par contre, si seulement un agent émet le symptôme et pas les autres, l'agent de diagnostic aura une confiance moindre quant au diagnostic qu'il va générer à partir de ce symptôme.

Dans l'architecture DIAMOND présentée sur le schéma de la figure 1.13, mettant en œuvre le concept multi-agents, le système global de surveillance (ou Monitoring) et de Diagnostic M&D est lui même composé de systèmes de M&D qui couvrent plusieurs systèmes de M&D plus petit. Le système global est donc découpé en plusieurs sous-ensembles, appelés domaines, surveillés par un système de M&D. Un système de M&D possède son agent de surveillance qui génère des symptômes lorsque le domaine surveillé par le système devient défaillant. Il possède aussi un agent facilitateur, un agent de résolution de conflits et un ou plusieurs agents de diagnostic. Ces agents de diagnostic mettent en œuvre des techniques différentes pour diagnostiquer les composants du domaine surveillé. Les différentes méthodes implémentées par les agents de diagnostic permettent la détection des conflits dans les résultats produits. C'est l'agent de résolution de conflits qui va alors établir le diagnostic final. Si l'agent de résolution de conflit ne parvient pas à produire une solution, il fait alors appel à l'agent facilitateur pour qu'il communique son problème à d'autres agents facilitateurs pour qu'ils le soumettent aux agents de résolutions de conflits d'autres domaines de même niveau. Il est alors assez simple de définir une hiérarchie des différents agents implémentant l'architecture DIAMOND. Cette hiérarchie lui confère un caractère plutôt décentralisé que distribué. En effet, cette architecture est similaire à une organisation en holons.

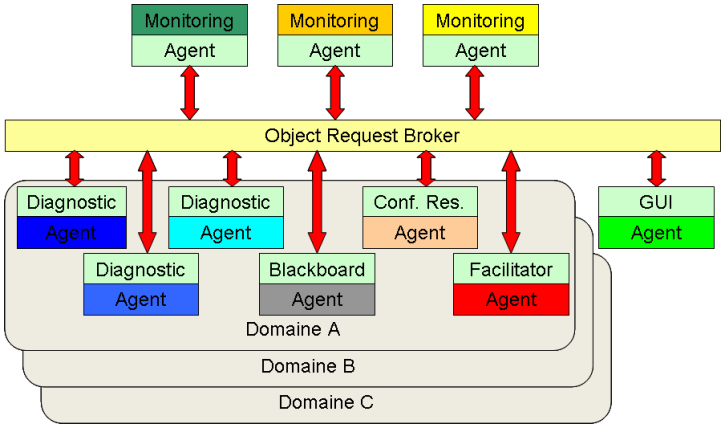


FIGURE 1.13: Architecture DIAMOND

Dans le projet DIAMOND (Albert *et al.*, 2002), les agents de diagnostic inscrivent leurs diagnostics locaux sur un tableau noir (blackboard). Un blackboard peut être assimilé à un espace mémoire partagé dans lequel les agents de diagnostic de l'architecture inscrivent leurs données et les rendent ainsi accessibles à tous les agents de diagnostic. Un agent facilitateur est en charge de gérer les conflits.

Dans cette architecture, les agents de diagnostic ne communiquent pas entre eux afin d'affiner leurs diagnostics locaux (gérer les conflits). Le blackboard assure la capitalisation des résultats de diagnostic et est géré par les agents de résolution de conflits. L'agent GUI (Graphical User Interface) assure l'interface entre le système DIAMOND et un utilisateur humain qui peut alors obtenir des informations sur l'état de l'équipement.

Différentes méthodes de diagnostic sont implémentées par le système DIAMOND permettant ainsi d'améliorer la fiabilité des résultats. Dans l'architecture DIAMOND, l'agent de résolution de conflits est très sollicité. Cependant, l'architecture est plutôt décentralisée. En effet, les agents de diagnostic ne communiquent pas entre eux directement puisque l'agent de résolution de conflits est chargé de l'élaboration du diagnostic correspondant à son niveau.

Des algorithmes de diagnostic distribué sont proposés par Hallgren et Skog (2005) et Biteus *et al.* (2008). Chaque agent local de diagnostic dispose d'un algorithme de diagnostic et d'un algorithme de résolution de conflits. Trois façons de raisonner sont alors proposées afin d'aboutir à un diagnostic global :

1. Soit extraire les conflits locaux, les corréler pour obtenir un conflit global puis en déterminer un diagnostic global. Cette méthode est assimilée à une approche centralisée. En effet, il y a un seul agent qui établit un diagnostic global.

2. Soit générer des diagnostics locaux à partir des conflits locaux et enfin définir un diagnostic global à partir de ces diagnostics locaux. Cette méthode est considérée comme distribuée par Biteus et hiérarchisée selon nous.

3. Soit de façon purement distribuée comme nous le définissons. L'objectif est de communiquer entre agents locaux par des échanges de conflits ou de diagnostics. Le sens de la transmission des conflits ou des diagnostics entre agents suit l'ordre inverse de la dépendance structurelle, c'est-à-dire du récepteur vers la source.

Dans notre cas, nous considérons les méthodes 1) et 2) comme étant décentralisées car un agent assure encore et toujours le diagnostic global du système. En remplaçant les entrées incriminées dans le diagnostic par les dépendances entre agents puis en extrapolant les variables des équations en termes de composants, un diagnostic est obtenu permettant d'incriminer uniquement des composants. Ce type de diagnostic permet donc à un agent d'incriminer un composant alors qu'il ne fait pas partie de son champ d'action. Un agent peut incriminer un composant surveillé par un autre agent. C'est cette manière de faire qui produit des conflits qu'il faut arbitrer par la suite. De plus, Biteus ne considère que deux modes de fonctionnement des composants. Soit ceux-ci fonctionnent de façon nominale et

n'apparaissent pas dans le diagnostic, soit ils sont définis comme défaillants par les agents de diagnostic.

Dans l'architecture WS-DIAMOND (Ardissono *et al.*, 2005), le système à diagnostiquer est découpé en plusieurs parties dont le diagnostic est établi par des agents de diagnostic locaux. Ces agents de diagnostic locaux n'ont aucune connaissance de la globalité du système mais uniquement de la partie qu'ils ont à diagnostiquer. Un superviseur de diagnostic détient quant à lui la connaissance des interactions entre chacunes des parties du système. C'est à lui qu'incombe la tâche de définir un diagnostic global du système à partir des diagnostics locaux. Cette technique est mise en œuvre par une architecture logicielle constituée exclusivement de web-services communicant entre eux. Cette architecture est donc décentralisée.

Dans une architecture de diagnostic WS-DIAMOND, il faut définir le découpage du système à diagnostiquer, et les Web-Services (WS). Ces WS sont dotés, en plus d'une fonction de diagnostic, leur conférant une capacité à s'auto-diagnostiquer, d'une fonction leur permettant de se reconfigurer eux-mêmes. Le but recherché est d'assurer la fiabilité, la disponibilité et plus généralement, la qualité de service ou QoS (Quality of Services) des WS fournis.

Le projet PROTEUS (Rebeuf *et al.*, 2004) avait pour objectif la définition d'une architecture distribuée de maintenance. L'implémentation proposée met en œuvre des WS facilitant l'évolution éventuelle du système de maintenance. L'architecture est basée sur une plate-forme sur laquelle les différentes applications de l'entreprise (ERP – Enterprise Resource Planning, base de données, portail web de l'entreprise, e-documentation, système de surveillance de paramètre (SCADA)...) sont connectées. Le système ne propose pas une architecture distribuée du diagnostic. La distribution est présente uniquement au niveau des connaissances et des fonctions de maintenance. En effet, celles-ci sont implémentées par des WS différents. Le système SCADA fait office de couche de surveillance. En effet, il fournit les informations à un WS de diagnostic qui transmet ensuite les siennes à un système d'aide à la décision pour la maintenance. Un ou plusieurs agents gèrent la base de connaissance nécessaire au diagnostic et à la prise de décision.

1.5. Pronostic distribué de systèmes

L'étude du pronostic, même s'il met en œuvre des méthodes issues des mêmes domaines que celles implémentées en diagnostic, commence seulement à se développer. Le nombre de références dans la littérature relative au pronostic distribué au sens auquel nous le concevons est très faible. Des premières propositions d'ar-

chitectures distribuées pour le pronostic ont été émises au début de l'année 2010. Les contributions présentées ci-dessous sont les principales références traitant du sujet du pronostic distribué même si le concept de la distribution qui y est développé est différent de l'approche proposée dans ce mémoire. Dans Saha *et al.* (2009) la distribution est abordée sous l'angle de la puissance de calcul. En effet, l'architecture composée de plusieurs agents pouvant communiquer entre eux par le biais d'une liaison filaire ou non commence par effectuer une tâche de diagnostic ne nécessitant pas de coopération entre les différents agents. Lorsqu'une faute est détectée, l'agent passe en mode pronostic et en avise la station de base. Celle-ci est en charge de planifier les tâches, de réinitialiser le processus en cas de détection d'erreurs et de fournir un accès aux ressources comme une base de données externe. Elle gère aussi la disponibilité des autres agents en termes de capacité de calcul et leur assigne alors une tâche de pronostic en plus de celle de diagnostic afin d'aider l'agent ayant détecté une faute dans sa tâche de pronostic. La station de base joue alors le rôle d'agent de coordination. Dragomir *et al.* (2007) présentent une évaluation de l'état de santé des systèmes à 2 niveaux : local, correspondant aux composants et global, associé au système. Pour cela, les agents locaux peuvent mettre en œuvre toutes les méthodes de pronostic disponibles en fonction de la connaissance sur le composant surveillé. Un agent global met en œuvre des réseaux de neurones pour évaluer l'état de santé du système global à partir des prédictions des agents locaux. Cette structure est donc purement décentralisée selon nous car le pronostic du système global est assuré par un unique agent. Une autre architecture distribuée de pronostic a été proposée par Takai et Kumar (2009). Cette architecture est composée de plusieurs agents locaux qui communiquent entre eux afin de converger vers un pronostic global du système surveillé. En effet, lorsqu'un évènement est observé par un agent local, il le transmet à tous les autres. Chacun des agents dispose alors de toutes les observations produites par l'ensemble des agents.

1.6. Problématiques abordées

D'après l'étude de la littérature que nous venons de présenter, nous pouvons dégager les quatre problèmatiques suivantes.

1.6.1. Évaluation de l'état de santé d'un STCGD

Évaluer l'état de santé d'un système revient à corréler les données de diagnostic et de pronostic afin de fournir à l'utilisateur une évaluation du système à l'instant

présent ou passé, ce qui revient à diagnostiquer le système et à fournir une tendance de son état futur fournie par la fonction pronostic. A cela des prévisions et des recommandations de remplacement datées sont ajoutées pour permettre à la maintenance de mettre en place une politique de CBM d'une part et de porter un avis sur la capacité du système à assurer ces missions futures d'autre part. Quoi qu'il en soit, toutes ces fonctions requièrent une grande quantité de connaissances qui, pour certaines, doivent être disponibles immédiatement dans le but d'être embarquées sur le système quand cela est possible. Actuellement, les architectures proposées ne traitent principalement que du diagnostic et donc de l'évaluation de l'état de santé actuel d'un système. Des méthodes de pronostic ont été proposées afin d'évaluer l'état de santé futur d'un composant mais jamais d'un système global. De plus, ces fonctions sont souvent déployées sur une architecture centralisée voire décentralisée. Le but de ce mémoire est de présenter une architecture distribuée de diagnostic et de pronostic permettant d'évaluer l'état de santé actuel et futur d'un STCGD dans sa globalité en partant de données de diagnostic et de pronostic générées au niveau des composants.

1.6.2. Limites des approches centralisées de diagnostic et de pronostic

Dans une architecture centralisée de diagnostic un seul et unique organe est chargé d'établir un diagnostic. Ceci pose évidemment des problèmes de fiabilité :

- Fiabilité au niveau du système en lui-même car si cet organe venait à tomber en panne, nous n'aurions plus la possibilité d'établir une liste de composants à remplacer.
- Fiabilité au niveau du résultat fourni par ce système.

Tout cela dépend évidemment de la taille du système à diagnostiquer. Si celui-ci est de petite taille, une architecture centralisée peut convenir. Si, par contre, le système est de grande taille et très complexe, ceci suppose une grande quantité d'informations à traiter et donc un risque de ne jamais converger vers la solution voire un risque d'effondrement du système de calcul. La limite de traitement du système pourrait être rapidement atteinte. Tout dépend aussi du temps dont dispose le système entre la réception de deux données. Si l'envoi de messages est synchrone et que le temps de traitement de l'information est inférieur au temps entre deux messages, le système aura tout le temps d'effectuer son traitement. Si par contre, le temps de traitement est supérieur au temps séparant la réception de deux messages, une situation de goulot en entrée du système de diagnostic peut se produire. La tâche est tout aussi compliquée quand l'envoi de messages est asynchrone. Une autre limite des approches centralisées concerne la mise à jour

et l'évolution de ce type d'architecture. En effet, lorsque le système à surveiller évolue, il faut faire évoluer le système de surveillance et remettre à jour tous les algorithmes de diagnostic.

1.6.3. Limites des approches décentralisées

Pour pallier le problème de charge de l'organe principal d'une architecture centralisée et ainsi pouvoir surveiller des systèmes fournissant des quantités d'informations supérieures, les architectures décentralisées peuvent être envisagées. Elles permettent de diminuer la quantité de données à traiter en la distribuant sur des agents de diagnostic locaux. Ceux-ci envoient ensuite leurs résultats à un agent de diagnostic global qui assure le diagnostic de l'ensemble du système. Si celui-ci venait à défaillir, la liste des composants à remplacer ne pourrait pas être établie. La mise à jour du système est simplifiée car il suffit de mettre à jour l'agent de diagnostic surveillant la partie du système qui a été modifiée en faisant évoluer un algorithme de taille moindre en comparaison à celui qu'aurait l'agent de diagnostic d'une architecture centralisée effectuant le diagnostic du même équipement.

1.6.4. Limites des approches distribuées

Mettre en place une approche distribuée permet de pallier le problème de la fiabilité du système. En effet, aucun organe n'est hiérarchiquement en charge d'établir un diagnostic global du système, ce qui implique qu'un diagnostic global existe toujours à moins que tous les agents locaux tombent en panne. Les agents locaux de diagnostic doivent communiquer entre eux afin de tendre localement et globalement à un diagnostic cohérent. Evidemment, ce type d'architecture tend à accroître la quantité de données échangées sur le bus de communication nécessaire non seulement à la collecte des informations issues de la surveillance du STCGD mais aussi aux échanges de messages entre les agents de l'architecture. Les questions à poser lorsqu'une architecture distribuée de diagnostic et de pronostic est envisagée sont : Comment fractionner la fonction diagnostic avant de la distribuer aux différents agents ? Comment être sûr de la convergence de la solution dans un système purement distribué ? Un découplage des fonctions détection et diagnostic/pronostic doit être également considéré afin, d'une part, de faciliter l'implémentation de la meilleure méthode de surveillance en fonction de la connaissance disponible de l'élément surveillé et, d'autre part, de rendre les fonctions de diagnostic et de pronostic génériques par la spécification des symptômes et des RUL fournis par les activités de surveillance. Après avoir présenté le contexte de l'étude et listé des avantages et inconvénients des approches centralisées et décentralisées de diagnos-

tic/pronostic, nous définissons les différentes contraintes émergentes. En effet, le donneur d'ordre tend à responsabiliser de plus en plus l'équipementier vis-à-vis des informations de l'état de leur équipement, car c'est à partir de ces informations qu'un diagnostic est établi. Ainsi, pour permettre un diagnostic pertinent, il faut que celui-ci soit basé sur des données elles aussi pertinentes. Il est donc de la responsabilité de l'équipementier de s'assurer de la validité des données transmises par son module, d'autant plus qu'il en connaît le modèle. La connaissance du modèle par le donneur d'ordre lui permettrait de réaliser plus facilement les fonctions de diagnostic. Cependant celui-ci aura beaucoup de mal à se procurer, sous réserve que l'équipement lui-même a un accès à la connaissance. Le problème réside aussi dans la capacité du système de diagnostic à pouvoir établir, à partir des symptômes détectés, un diagnostic unitaire n'incriminant qu'un seul composant dans un système présentant de nombreuses interactions. Il doit aussi pouvoir déterminer la fonction défaillante du STCGD. L'ajout d'une fonction pronostic exploitant les résultats de diagnostic doit permettre l'évaluation de l'aptitude de la ressource à répondre aux futurs engagements.

1.7. Conclusion

Nous avons établi qu'une contribution de ce mémoire doit être de proposer une architecture visant à évaluer l'état de santé des STCGD. Ceux-ci sont de plus en plus complexes car devant répondre à des exigences de certifications et de sureté de fonctionnement normalisées de plus en plus sévères. L'augmentation du nombre de services proposés par ces systèmes complexifie leur mise en œuvre et augmente le nombre de données échangées.

L'arrivée des NTIC a eu un impact significatif sur les architectures actuellement développées. Ils permettent des déploiements autres que centralisés des fonctions de diagnostic et de pronostic mais aussi de fiabiliser et d'augmenter la capacité de communication en termes de flux de données. Les architectures centralisées conduisent à la non-répartition des données facilitant ainsi la mise à jour de la base de connaissances. Les architectures décentralisées, quant à elles, visent à améliorer la fiabilité du système et permettent un développement des algorithmes de surveillance et de diagnostic/pronostic propres à la connaissance de la fonction surveillée. L'adjonction à la fonction de diagnostic d'une fonction de pronostic permet de définir la capacité du système technique à accomplir son ou ses futurs engagements dans des objectifs préétablis. L'ensemble des données ainsi générées vise à évaluer l'état de santé actuel et futur du système et propose un support d'aide

à la décision pour la mise en place d'une politique de maintenance préventive conditionnelle optimisant l'utilisation des composants. Il apparait donc que le diagnostic et le pronostic sont deux activités présentant un enjeu majeur pour l'exploitation des systèmes et pour l'évaluation de leur état de santé. Toutefois le développement d'algorithmes permettant de définir l'état de santé global de systèmes demeure un sujet ouvert. Dans ce premier chapitre, nous avons discuté les techniques mises en œuvre dans des problématiques de surveillance-diagnostic et défini les concepts de base afférants aux différentes problématiques. Le deuxième chapitre est dédié à l'analyse des fonctions de diagnostic et de pronostic qui exploitent les connaissances disponibles du système. Une définition de la connaissance minimale nécessaire aux fonctions de diagnostic et de pronostic permet d'appréhender la complexité d'un système à étudier. Par la suite, les principes de diagnostic et de pronostic exploitant les connaissances modélisées sont exposés.

Chapitre 2

Analyse et modélisation pour l'évaluation de la santé de systèmes techniques complexes de grande dimension

2.1. Introduction

Nous avons défini la catégorie des Systèmes Techniques Complexes de Grande Dimension (STCGD) et les contraintes qui leurs sont inhérentes nécessitant notamment la mise en place d'une maintenance préventive conditionnelle. Cette politique de maintenance permet, entre autres apports, d'allonger la durée d'utilisation des composants du système technique ou d'espacer les entretiens tout en conservant la possibilité de planification des interventions. Les fonctions de surveillance, diagnostic et pronostic de systèmes techniques sont indispensables à la mise en place d'une maintenance préventive conditionnelle. La fonction de surveillance permet de fournir les informations exploitables par les fonctions de diagnostic et de pronostic à partir de données prélevées sur le système technique.

Dans ce chapitre, nous analysons les fonctions de diagnostic et de pronostic dans le cas d'applications à des STCGD. Il a pour but de définir :

- les entrées de ces deux fonctions dont certaines sont produites par la fonction de surveillance,
- les résultats produits par les fonctions de diagnostic et de pronostic,
- les informations et/ou les connaissances supports aux traitements réalisés par ces fonctions ainsi que leur modélisation afin de les exploiter.

Avant de conclure ce chapitre, une identification des statuts que peut prendre un composant est présentée.

2.2. Formalisation d'un STCGD

Comme nous l'avons vu à la section 1.2.3, un STCGD peut être décomposé en plusieurs sous-parties jusqu'au niveau composant. Le critère d'arrêt de la décomposition est le composant remplaçable (ou réparable). Comme nous l'avons établi dans le chapitre 1, un STCGD intègre un certain nombre de systèmes. Ainsi,

un STCGD sera considéré comme étant composé de n systèmes physiques SP. STCGD $= \{SP_1, SP_2 \ldots SP_n\}$. Par exemple, aujourd'hui, un avion est composé d'environ 80 systèmes. Chaque SP est composé de m fonctions F. Pour visualiser cette information, l'indice du SP est conservé. Ce qui donne :

$SP_i = \{F_{i,1}, F_{i,2} \ldots F_{i,m}\}$ avec $F_{i,j}$ désignant la $j$ème fonction du $i$ème SP. Le même principe est utilisé ensuite pour les composants et les symptômes. Ainsi, une fonction $F_{i,j}$ est implémentée par p composants, ce qui donne :

$F_{i,j} = \{CR_{i,j,1}, CR_{i,j,2} ... CR_{i,j,p}\}$ avec $CR_{i,j,k}$ désignant le composant numéro k de la fonction $F_{i,j}$. De même,$S_{i,j,k,l}$ désignera le symptôme$S$ numéro l pouvant être généré par la fonction de génération de symptôme surveillant le composant $CR_{i,j,k}$.

2.3. Analyse de la fonction de diagnostic

Pour établir un diagnostic les trois fonctions suivantes sont généralement mises en œuvre : détection, isolation et identification. Les approches de diagnostic développées actuellement par la communauté de l'automatique sont focalisées essentiellement sur les techniques de génération et de structuration des résidus ou sur des représentations multi-modèles (Niemann *et al.*, 2007). D'une façon plus générale, Zemouri (2003) et Chiang *et al.* (2001) exposent différentes méthodes de diagnostic :

- basées sur des modèles,
- basées sur des données (PCA, FDA, PLS, CVA),
- analytiques (observateurs, relations de parité),
- basées sur la connaissance (analyse causale, systèmes experts, reconnaissance de formes, systèmes hybride).

De nombreux travaux ont été effectués dans le domaine de la détection et de l'isolation de fautes (FDI) (Isermann, 1997; Glavaski et Elgersma, 2001; Halder *et al.*, 2004), les outils développés sont souvent issus du domaine fiabiliste (Papadopoulos, 2003; Wu et Campion, 2004) ou encore de l'Intelligence Artificielle (IA) (Liu, 1996; Chen et Lee, 2002). Les méthodes mises en œuvre pour le diagnostic d'un système suivent les méthodes implémentées pour la détection. Quoiqu'il en soit, le diagnostic vise à isoler et à identifier la cause d'une défaillance, par exemple, la faute ayant conduit la couche de détection à générer un symptôme. Un symptôme est une preuve observable d'une défaillance sensorielle ; c'est-à-dire un changement d'une quantité observable du comportement normal spécifié (Isermann, 1997). Pour confirmer ou infirmer le résultat de la fonction diagnostic, il est possible de générer

des tests qui permettent de reproduire des situations opérationnelles du système afin de recréer la défaillance fonctionnelle. Ces générations de tests peuvent être effectuées de façon automatique (la fonction diagnostic peut lancer elle-même le test en fonction du résultat de diagnostic obtenu) ou semi-automatique (un agent de maintenance lance manuellement le test). Dans certains cas comme les avions, des tests ne peuvent en effet être déclenchés qu'une fois que le système a achevé sa mission (comme la sortie des aérofreins pour les avions). Dans le cas des STCGD, le diagnostic vise à déterminer le composant en faute ayant entrainé une défaillance fonctionnelle à partir d'une surveillance au niveau des composants. Le but d'une fonction de diagnostic est l'identification d'un ensemble $L2$ de fonctions défaillantes et la localisation de leurs causes qui peut être un ensemble $L1$ de composants en faute à partir d'un ensemble de symptômes S et d'un ensemble de tests T. Cela conduit à la relation suivante :

$$(L1, L2) = Diag(SKDiag, S, T)$$

où $Diag$ est la fonction de diagnostic et $SKDiag$ désigne la connaissance disponible sur le système support de la fonction de diagnostic. Cette connaissance est présentée dans la section 2.5.

La fonction $Diag$ peut être implémentée grâce à 2 sous-fonctions $Diag1$ et $Diag2$ selon une approche ascendante (des composants vers les fonctions). La fonction $Diag1$ identifie à partir d'un ensemble de symptômes et d'un ensemble de tests l'ensemble des composants en faute du système :

$$L1 = Diag1(SKDiag, S, T)$$

où $L1 = \{AB(CR_{i,j,k}), ..., AB(CR_{p,r,s})\}$

et la fonction $Diag2$ identifie l'ensemble des fonctions défaillantes à partir de tests et des CR en fautes :

$$L2 = Diag2(SKDiag, L1, T)$$

où $L2 = \{AB(F_{i,j}), ..., AB(F_{p,r})\}$

La notation $AB(.)$ dénote aussi bien la défaillance d'une fonction désignée par $F_{i,j}$ que la faute d'un composant désigné par $CR_{i,j,k}$. L'ensemble de la fonction de diagnostic est présentée sous la forme d'un diagramme IDEF0 sur la figure2.1.

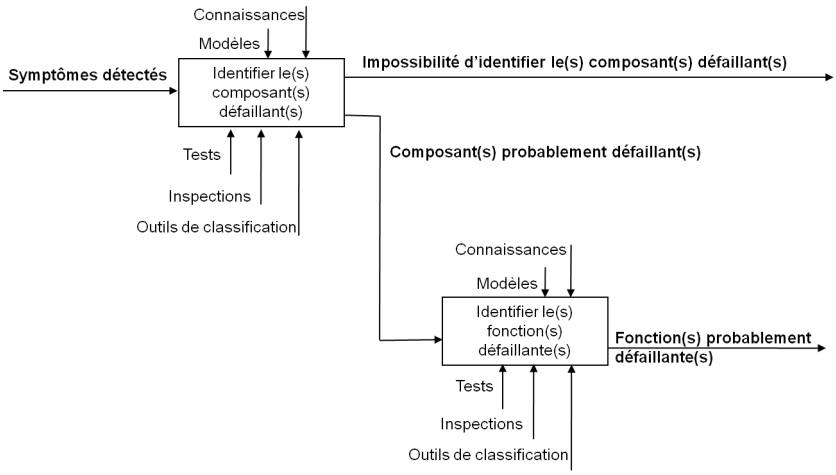


FIGURE 2.1: IDEF0 de la fonction diagnostic

2.4. Analyse de la fonction de pronostic

Les méthodologies de pronostic peuvent être divisées en trois catégories :

- celles basées sur un modèle physique,
- celles guidées par les données,
- celles basées sur l'expérience.

Le choix d'utilisation d'une de ces méthodologies est fonction des données disponibles qui peuvent être fournies en entrée de la fonction de pronostic (Muller, 2005). Elles sont cependant issues des même domaines de recherche que les fonctions de diagnostic : approches multi-modèles (Luo *et al.*, 2003; Byington *et al.*, 2004b), approches fiabilistes (Wilkinson *et al.*, 2004) ou approches basées sur l'IA (Sakellaropoulos et Nikiforidis, 2000)... Kothamasu *et al.* (2006) proposent une liste de méthodes pouvant être implémentées pour assurer le pronostic et le suivi de l'évolution d'une dégradation d'un système. Chacune d'entre elles est l'objet d'une description sommaire dans les parties suivantes (Vachtsevanos *et al.*, 2006). Quoi qu'il en soit une fonction pronostic vise à définir la durée de vie résiduelle d'un composant (RemainingUseful Life - RUL) ou bien encore la probabilité qu'une défaillance intervienne avant un instant donné. Le RUL peut être défini comme donnant le temps avant une probabilité de défaillance devenant inacceptable au regard des risques encourus par les biens, les personnes et l'environnement. Dans la suite du mémoire, le RUL est défini comme étant le temps avant que le composant surveillé ne provoque une défaillance fonctionnelle à partir des conditions opérationnelles (ou contraintes). Cela signifie que la fonction de pronostic requiert

un historique des missions mais aussi des futures missions que le système aura à assurer ainsi que leurs contraintes. Le RUL d'un composant, lui-même noté CR, peut être calculé par la couche de surveillance qui va générer des RUL et les envoyer à la fonction de pronostic. L'idéal serait que le calcul du RUL soit assuré par l'équipementier à partir de modèles bien calés dont il est le seul à avoir la connaissance. Dans la pratique, des RUL sont fournis pour les composants mais la valeur est généralement donnée avec une marge de sécurité confortable. Les exploitants souhaitent maximiser les temps d'engagement et disposer d'une information liée à l'utilisation du composant. Nous notons $RUL(CR_{i,j,k})$ le RUL du $CR_{i,j,k}$. Nous introduisons la fonction $TBAB(CR_{i,j,k})$ correspondant au temps restant avant qu'un comportement anormal (Time Before Abnormal Behavior) pour le $CR_{i,j,k}$ ne se produise pour une raison pouvant être extérieure au CR. La fonction $TBAB(F_{i,j})$ est également introduite pour une fonction $F_{i,j}$. pour tenir compte du fait qu'une fonction est mise en œuvre par un ensemble de CR. En effet, le RUL est une donnée générée par la couche de surveillance du composant, le TBAB a été introduit pour tenir compte des dépendances entre les CR. Le calcul des $TBAB$ par la fonction de pronostic peut être définie par :

$$TBAB(CR_{i,j,k}) = Min(RUL(CR_{i,j,k}), TBAB(CR_{i,j,k}), TBAB_{struc}(CRs))$$

où $TBAB_{struc}(CRs)$ désigne l'ensemble des $TBAB$ des CR en amont de $CR_{i,j,k}$ structurellement parlant. Par exemple, considérons une unité de calcul ayant un RUL de 10 Unités de Temps (UT). Cependant, pour fonctionner elle a besoin d'une alimentation électrique composée d'un onduleur avec un RUL de 6 UT et d'un groupe électrogène avec un RUL de 12 UT. Dans ce cas, $TBAB$(onduleur) vaut 6 UT, mais $TBAB$(unité de calcul) ne vaut que 6 UT. Ainsi, pour une fonction $F_{i,j}$ donnée sans CR redondants :

$$TBAB(F_{i,j}) = Min(TBAB(CR_{i,j,1}), ..., TBAB(CR_{i,j,k}))$$

En considérant le cas où la fonction $F_{i,j}$ est implémentée par N CRs redondants et que cette fonction n'est pas considérée comme défaillante, si au moins M de ces N CR ne sont pas défaillants alors $TBAB(F_{i,j})$ est égale au Mième $TBAB$ le plus élevé parmi les N $TBAB$ de chacun des CR implémentant $F_{i,j}$. Par exemple, si une fonction $F_{i,j}$ est assurée par $CR_{i,j,1}$, $CR_{i,j,2}$ et $CR_{i,j,3}$, et que la fonction n'est pas considérée comme défaillante si au moins 2 de ses 3 CR ne sont pas défaillants.

Avec $TBAB(CR_{i,j,1})$= 5 UT, $TBAB(CR_{i,j,2})$ = 10 UT et $TBAB(CR_{i,j,3})$ = 15 UT, alors $TBAB(F_{i,j})$ = 10 UT.

Le pronostic identifie un ensemble $H2$ de $TBAB$ de chaque fonction implémentant le système et un ensemble $H1$ de $TBAB$ de chaque CR à partir d'un ensemble de RUL. Cela conduit à la relation suivante où $Prog$ est la fonction de pronostic :

$$(H1, H2) = Prog(SKProg, RUL)$$

où $SKProg$ désigne la connaissance disponible sur le système support à la fonction de pronostic. La fonction $Prog$ peut être implémentée grâce à 2 sous-fonctions $Prog1$ et $Prog2$. La fonction $Prog1$ permet, à partir d'un ensemble de RUL, d'identifier l'ensemble des $TBAB$ des CR d'une fonction :

$$H1 = Prog1(SKProg, RUL)$$

où $H1 = \{TBAB(CR_{i,j,k}), ..., TBAB(CR_{p,r,s})\}$ et la fonction $Prog2$ permet d'identifier l'ensemble des $TBAB$ des fonction du système à partir des $TBAB$ des CR et de la modélisation fonctionnelle :

$$H2 = Prog2(SKProg, H1)$$

où $H2 = \{TBAB(F_{i,j}), ..., TBAB(F_{p,r})\}$. $TBAB(.)$ définit aussi bien le $TBAB$ d'un CR que celui d'une fonction. L'ensemble de la fonction de pronostic est présentée sous la forme d'un diagramme IDEF0 sur la figure 2.2.

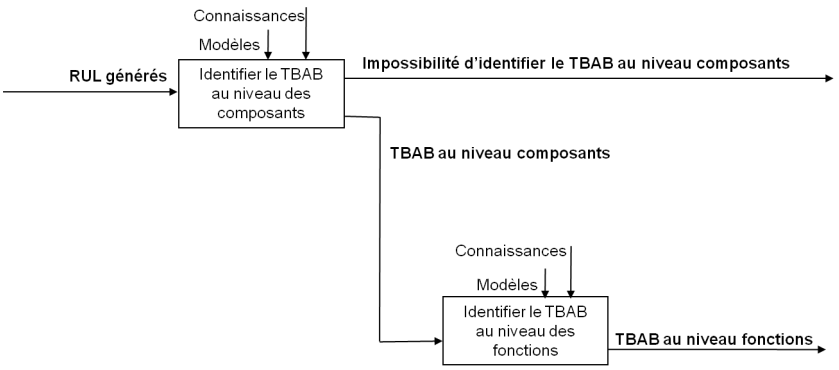


FIGURE 2.2: IDEF0 de la fonction pronostic

2.5. Connaissances pour le diagnostic et le pronostic

Durant la conception du système, les différentes études menées afin d'implémenter un STCGD produisent des informations ou connaissances utiles au développement des fonctions de diagnostic et de pronostic de ce même STCGD. Plusieurs catégories de connaissances alors rendues disponibles sont nécessaires (Reiter, 1992; Chittaro *et al.*, 1993) :

- La connaissance structurelle vise à représenter l'ensemble des composants implémentant le système ainsi que leurs interconnexions. Elle est utilisée dans les travaux de Worn *et al.* (2004), Biteus (2005) et Touaf (2005) et a surtout permis de raffiner le diagnostic en propageant les défaillances entre composants interconnectés. Dans (Ardissono *et al.*, 2005), la connaissance structurelle, étant donné que nous sommes en présence de web-services, porte sur la composition des web-services, c'est-à-dire des appels de web-services par des web-services.
- La connaissance fonctionnelle vise à représenter les composants concourant à l'implémentation des différentes fonctions du STCGD. Cette modélisation est basée sur la connaissance issue de la description structurelle car elle regroupe les composants en fonctions explicites pour l'utilisateur (pilote, conducteur, opérateur...) du STCGD. Certains composants peuvent contribués à l'implémentation de plusieurs fonctions.
- La connaissance topologique vise à représenter les influences par effet de proximité que les composants peuvent avoir entre eux comme, par exemple, des interactions thermiques, électromagnétiques...
- La connaissance comportementale vise à représenter la dynamique du composant étudié afin de déterminer le comportement global du système. Les modèles sont utilisés principalement pour la détection de symptômes (modèles dynamiques continus, modèles dynamiques échantillonnés, modèles dynamiques à événements discrets ou modèles hybrides). Ces modèles peuvent aussi bien décrire les différents modes de fonctionnement de composants ou de fonctions du STCGD dont des modes de défaillances. Cette description porte aussi sur les liens de causalité entre composants. La connaissance comportementale contient aussi les modèles de bon et de mauvais fonctionnement dans le cas de diagnostic multi-modèles (Worn *et al.*, 2004; Ardissono *et al.*, 2005; Biteus, 2005; Touaf, 2005). Comme nous l'avons mentionné dans la section 1.2.3, nous nous intéressons surtout aux modèles de mauvais fonctionnement et donc à la définition des relations entre défaillances et symptômes.

D'autres types de connaissances présentent également un intérêt dans la mise en

œuvre de fonctions de diagnostic et de pronostic, comme les redondances conçues pour répondre à des critères de fiabilité, et les arbres de défaillances qui peuvent être construits à partir des connaissances issues de la conception du STCGD. Cependant, ces connaissances doivent être modélisées afin d'être rendues exploitables par les fonctions de diagnostic et de pronostic.

2.5.1. Représentation de la connaissance structurelle

2.5.1.1. Définition

La connaissance structurelle MS vise à représenter l'ensemble des composants implémentant le système ainsi que leurs interconnexions.

2.5.1.2. Exemple

Soit une fonction implémentée par l'interconnexion de 5 composants remplaçables $CR_{1,1,1}$, $CR_{1,1,2}$, ..., $CR_{1,1,5}$ comme cela est représenté sur la figure 2.3. Chacune des entrées (input I) et sorties (Output O) des composants seront numérotées (I_1,I_2, ... et O_1, O_2,...).

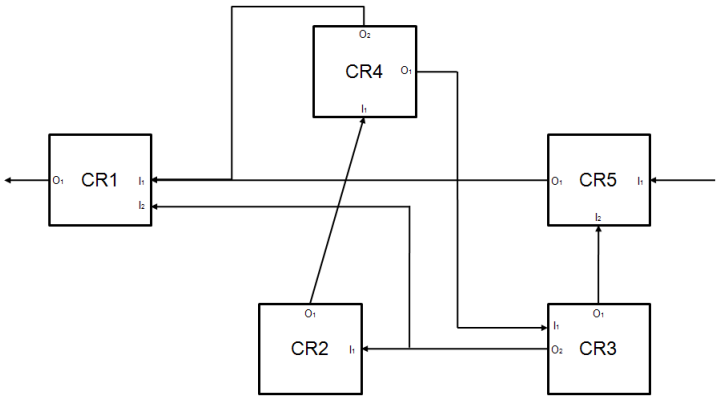


FIGURE 2.3: Exemple de fonction pour une description structurelle

2.5.1.3. Formalisation

Pour modéliser ces interconnexions, nous considérons $O_x(CR_{i,j,k})$ représentant la sortie numéro x du composant $CR_{i,j,k}$ et $I_y(CR_{i,j,k})$ l'entrée numéro y du composant $CR_{i,j,k}$ avec $x \in [1,r]$, $y \in [1,s]$, $n \in [1,m]$, $(r,s) = f(CR_{i,j,k})$ où f est la fonction implémentée par le CR, m désigne le nombre maximum de composants implémentant le système. En appliquant, ces notations dans le cas de l'exemple de la figure 2.3, les relations suivantes sont obtenues : $O_1(CR_{1,1,1}) = I_2(CR_{1,1,5})$;

$O_1(CR_{1,1,5}) = I_1(CR_{1,1,1})$; $O_1(CR_{1,1,4}) = I_1(CR_{1,1,3})$; $O_2(CR_{1,1,3}) = I_1(CR_{1,1,2})$; $O_2(CR_{1,1,3}) = I_2(CR_{1,1,1})$; $O_2(CR_{1,1,4}) = I_1(CR_{1,1,1})$. Elles signifient que la sortie O_1 du composant $CR_{1,1,1}$ est reliée à l'entrée I_2 du composant $CR_{1,1,5}$... La seule exception venant bien évidemment de $I_1(CR_{1,1,5})$ et de $O_1(CR_{1,1,1})$ qui désigne l'entrée et la sortie uniques de la fonction $F_{1,1}$ implémentée par ces cinq composants. Une fonction peut bien évidemment avoir plusieurs entrées et plusieurs sorties.

Hormis les entrées et sorties d'une fonction implémentée par des composants, nous prendrons comme hypothèse qu'il y a toujours une sortie connectée à une entrée ($O = I$). Des équations du type $O = O$, $I = I$ et $I = O$ ne peuvent donc pas être définies.

Dans les travaux présentés dans ce mémoire, il n'y a pas d'intérêt à détailler les liaisons entre les entrées et les sorties des différents composants. La modélisation structurelle détermine ainsi les connexions entre les composants comme cela est représenté le schéma de la figure 2.4.

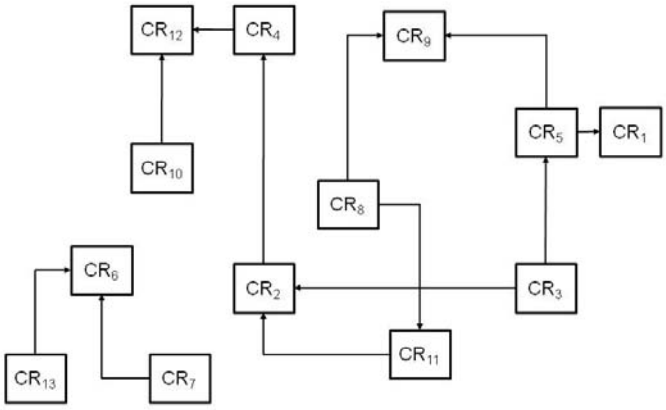


FIGURE 2.4: Modélisation structurelle

Nous avons également considéré que la relation qui lie un composant à un autre n'est pas uniquement un lien physique entre 2 composants mais aussi une relation pour laquelle une défaillance d'un composant peut engendrer la défaillance d'un autre composant. La défaillance d'un composant étant due à une faute observée par le biais d'au moins un symptôme généré par la fonction de surveillance. Un

composant peut être l'objet de défaillances de natures différentes. Ces dernières peuvent être observées par des symptômes différents. Il est alors préférable de définir une liaison entre symptômes de composants différents structurellement liés. En effet, la défaillance d'un composant peut avoir ou ne pas avoir de conséquence sur le fonctionnement d'un autre composant qui lui est structurellement lié. Une liaison entre symptômes de composants différents peut donc être considérée même si ces composants participent à l'implémentation d'une même fonction. Ces relations sont des relations orientées de couples de symptômes. Pour représenter cette connaissance et la rendre exploitable par les fonctions de diagnostic et de pronostic, nous avons choisi de les formaliser dans des fichiers au format XML (eXtensible Markup Language) plutôt qu'une autre implémentation pour les raisons suivantes (Lebold *et al.*, 2002) :

- il est possible de donner le nom que l'on veut aux balises permettant de délimiter les données que le fichier contient ;
- les données sont stockées sous la forme d'une arborescence qui correspond, pour certaines, à la structure de la connaissance décrite ;
- comme le format XML privilégie le contenu plutôt que la présentation des données, il est possible de représenter et trier les données en fonction du métier qu'exerce la personne (opérateur de maintenance ou de conduite) en utilisant les technologies appropriées (CSS...) ;
- il est possible de vérifier la bonne construction du document par des fichiers permettant de définir les noms et le nombre de balises présentes dans le document.

La formalisation de la connaissance structurelle dans un fichier XML dont la structure des enregistrements est décrite sous la forme d'un schéma XML représentée sur la figure 2.5.

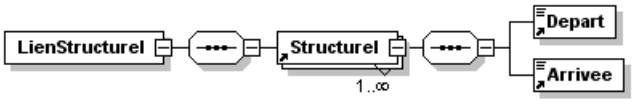


FIGURE 2.5: Schéma XML de la connaissance structurelle

En notant alors S_i le symptôme numéro i pouvant être généré par la fonction de surveillance et révélant un dysfonctionnement d'un composant noté CR. Un lien structurel entre 2 composants est alors défini par une relation de cause à effet

entre 2 symptômes. L'agrégat $Struct(S_{i,j,k,l}, S_{a,b,c,d})$ définit alors un lien structurel entre le symptôme $S_{i,j,k,l}$ de $CR_{i,j,k}$ et le symptôme $S_{a,b,c,d}$ de $CR_{a,b,c}$.

2.5.2. Représentation de la connaissance fonctionnelle

2.5.2.1. Définition

La connaissance fonctionnelle MF vise à représenter les composants concourant à l'implémentation des différentes fonctions du STCGD. Cette modélisation est basée sur la connaissance issue de la description structurelle car elle regroupe les composants en fonctions explicites pour l'utilisateur (pilote, conducteur, opérateur...) du STCGD. Certains composants peuvent être utilisés à l'implémentation de plusieurs fonctions.

2.5.2.2. Exemple

Soit un système implémenté par 4 fonctions$F_{1,1}$, $F_{1,2}$, $F_{1,3}$ et $F_{1,4}$ différentes liées fonctionnellement comme cela est représenté sur le schéma de la figure 2.6.

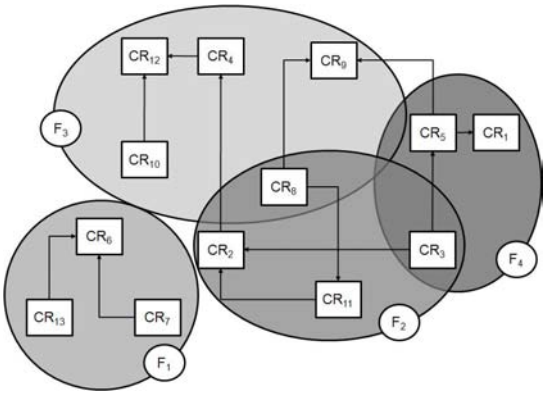


FIGURE 2.6: Exemple de système pour la description fonctionnelle

2.5.2.3. Formalisation

La modélisation fonctionnelle vise à constituer des ensembles de composants correspondants aux différentes fonctions du point de vue des utilisateurs du STCGD. Un exemple d'analyse d'un STCGD par ses fonctions vues des utilisateurs est représenté sur le diagramme de classe UML (Unified Modeling Language) de la figure 2.7. Le STCGD est structuré en plusieurs fonctions implémentées chacune par plusieurs composants. Un composant peut contribuer à plusieurs fonctions du STCGD.

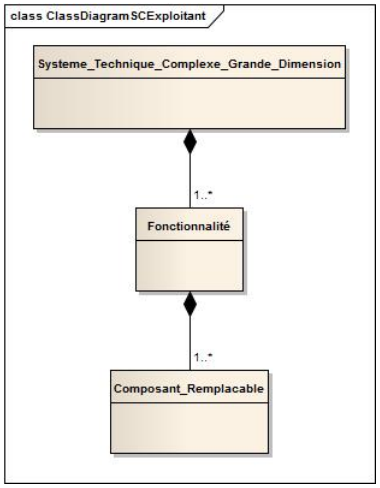


FIGURE 2.7: Découpage fonctionnelle

Nous considérons, comme sur le diagramme de la figure 2.7, deux niveaux de découpage : le niveau des composants et le niveau des fonctions. Nous notons $F_{x,y}$ la fonction numéro y du système numéro x avec $y \in [1, t]$, t désignant le nombre total de fonctions du système y du point de vue des utilisateurs du STCGD. $CR_{x,y,z}$ représente le composant numéro n avec $z \in [1, N]$, N désignant le nombre total de CR du STCGD.

En appliquant ces notations à l'exemple de la figure 2.6, les relations suivantes sont obtenues :

- $F_{1,1} = CR_{1,1,6}, CR_{1,1,7}, CR_{1,1,13}$
- $F_{1,2} = CR_{1,2,2}, CR_{1,2,3}, CR_{1,2,8}, CR_{1,2,11}$
- $F_{1,3} = CR_{1,3,4}, CR_{1,3,8}, CR_{1,3,9}, CR_{1,3,10}, CR_{1,3,12}$
- $F_{1,4} = CR_{1,4,1}, CR_{1,4,3}, CR_{1,4,5}$

Ces ensembles peuvent être constitués lors de la conception du STCGD. Ces relations fonctionnelles représentées par appartenances à des ensembles peuvent être rendues exploitables par les fonctions de diagnostic et de pronostic à l'aide de fichier XML dont la structure des enregistrements traduisant ces appartenances est décrite sous la forme du schéma XML de la figure 2.8.

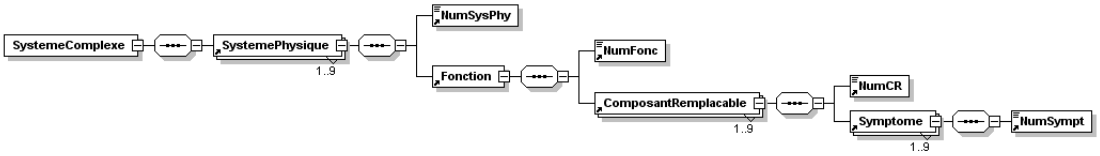


FIGURE 2.8: Schéma XML de l'arborescence d'un système complexe

2.5.3. Représentation de la connaissance topologique

2.5.3.1. Définition

La connaissance topologique MT vise à représenter les influences par effet de proximité que les composants peuvent avoir entre eux comme, par exemple, des interactions thermiques, électromagnétiques...

2.5.3.2. Exemple

Nous considérons deux composants sachant que seulement certains types de défaillances peuvent engendrer une dépendance topologique. Si deux composants, notés $CR_{1,1,1}$ et $CR_{1,1,2}$, sont liés topologiquement la relation est notée : $CR_{1,1,1} \rightarrow CR_{1,1,2}$.

Cette relation est unidirectionnelle. En effet, en considérant la représentation de la figure 2.9 de deux composants $CR_{1,1,1}$ et $CR_{1,1,2}$ pour laquelle $CR_{1,1,1}$ est spatialement situé en dessous de $CR_{1,1,2}$, que $CR_{1,1,1}$ est un composant principalement constitué d'électronique alors que $CR_{1,1,2}$ est un distributeur hydraulique, dans le cas où le distributeur est grippé, cela n'a aucune influence sur $CR_{1,1,1}$. Par contre, si $CR_{1,1,2}$ fuit, par gravité, les gouttes tomberont sur $CR_{1,1,1}$ et pourront provoquer un court-circuit. Si $CR_{1,1,2}$ devenait défaillant, comme par exemple se mettre à chauffer, cela n'aurait aucun impact sur $CR_{1,1,1}$. Cet exemple est une illustration du choix qui a été fait d'avoir des dépendances topologiques unidirectionnelles. Le lien topologique ne concerne que certaines défaillances de composants (une fuite dans notre exemple) et qu'elle est valable uniquement dans ce cas et dans cette configuration du système ($CR_{1,1,2}$ au-dessus de $CR_{1,1,1}$) sinon, la fuite n'aurait aucune influence sur $CR_{1,1,1}$.

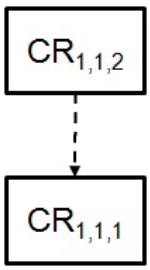


FIGURE 2.9: Exemple de dépendance topologique

Soit S_i le symptôme numéro i pouvant être généré par la fonction de surveillance et révélant une défaillance particulière du composant. Nous définissons finalement un lien topologique entre deux composants par une relation de cause à effet entre 2 symptômes par l'agrégat $Topo(S_{i,j,k,l}, S_{a,b,c,d})$ définissant un lien topologique entre le symptôme $S_{i,j,k,l}$ de $CR_{i,j,k}$ et le symptôme $S_{a,b,c,d}$ de $CR_{a,b,c}$.

2.5.3.3. Formalisation

La modélisation de la connaissance topologique définit les effets de proximité entre des défaillances de composants. Dans cette modélisation, seule l'influence indirecte que peut avoir une défaillance d'un composant sur un autre est considérée même s'ils ne sont pas structurellement liés. En reprenant l'exemple illustré sur la figure 2.6 et en y ajoutant des relations de dépendances topologiques, nous obtenons l'exemple illustré sur la figure 2.10.

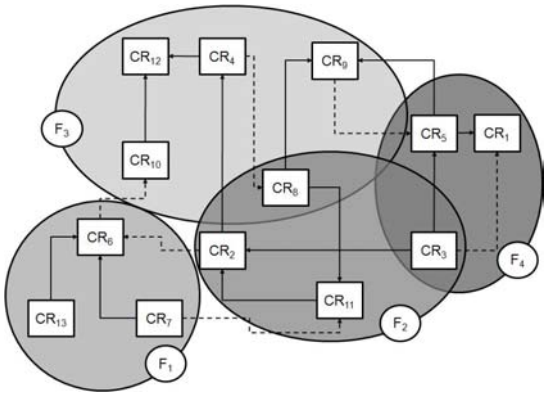


FIGURE 2.10: Modélisation topologique

Ainsi, par cette analyse, les composants contenus dans les sous-ensembles :
$\{CR_{1,1,6}, CR_{1,1,7}, CR_{1,1,13}\}$, $\{CR_{1,2,2}, CR_{1,2,3}, CR_{1,2,8}, CR_{1,2,11}\}$,
$\{CR_{1,3,4}, CR_{1,3,8}, CR_{1,3,9}, CR_{1,3,10}, CR_{1,3,12}\}$ et $\{CR_{1,4,1}, CR_{1,4,3}, CR_{1,4,5}\}$ peuvent avoir un effet les uns sur les autres. Comme seulement certains types de défaillances ou dysfonctionnements (échauffement, fuite...) peuvent avoir des conséquences sur l'état de santé de composants voisins, il est nécessaire d'analyser les défaillances et dysfonctionnements que peuvent subir les différents composants pour pouvoir établir des relations topologiques. Cette analyse, lorsqu'elle est menée en phase de conception du STCGD, peut remettre en cause la localisation des composants notamment pour des raisons de sécurité. Un objectif de cette analyse peut donc viser des conceptions de STCGD où la répartition spatiale des composants tend à

minimiser le nombre de relations topologiques entre les composants. Ces relations topologiques orientées peuvent être rendues exploitables par les fonctions de diagnostic et de pronostic par un fichier XML dont la structure des enregistrements est décrite sous la forme du Schéma XML de la figure 2.11.

FIGURE 2.11: Schéma XML de la connaissance topologique

2.5.4. Représentation de la connaissance comportementale

2.5.4.1. Définition

La modélisation de la connaissance comportementale MC vise à représenter la dynamique du composant étudié afin de déterminer le comportement global du système. Les modèles sont utilisés principalement pour la détection de symptômes (modèles dynamiques continus, modèles dynamiques échantillonnés, modèles dynamiques à événements discrets ou modèles dynamiques hybrides). Ces modèles peuvent aussi bien décrire les différents modes de fonctionnements de composants, de fonctions, du STCGD dont des modes de défaillances. Cette description porte aussi sur les liens de causalité entre composants. La connaissance comportementale contient aussi tous les modèles de bon et de mauvais fonctionnement dans le cas de diagnostic multi-modèles. Comme nous l'avons vu dans la section 1.2.3, nous nous intéressons surtout aux modèles de mauvais fonctionnement et donc à la définition des relations entre défaillances et symptômes.

2.5.4.2. Exemple

Prenons par exemple le cas d'un composant $CR_{1,1,1}$ à partir duquel il est possible de détecter 5 symptômes différents ($S_{1,1,1,1}$, $S_{1,1,1,2}$, $S_{1,1,1,3}$, $S_{1,1,1,4}$ et $S_{1,1,1,5}$). Il est possible alors de chercher à lier un ou des symptômes à des défaillances connues $Fuite = \{S_{1,1,1,1}, S_{1,1,1,3}, S_{1,1,1,5}\}$ ce qui signifie que si $S_{1,1,1,1}$, $S_{1,1,1,3}$ et $S_{1,1,1,5}$ apparaissent, alors $CR_{1,1,1}$ fuit. Un symptôme peut apparaître dans plusieurs défaillances (il est possible qu'un vérin ne coulisse pas lorsque cela est demandé soit parce qu'il est faussé, soit parce que l'électrovanne de commande n'est plus alimentée...). Une défaillance peut avoir pour origine une seule et unique faute

visible par un unique symptôme. Un symptôme ou un ensemble de symptômes ne correspondant pas à une signature de défaillance connue lors du développement du système de diagnostic, sera considéré comme entraînant la défaillance du composant. Cependant, la nature de cette défaillance ne pourra être donnée par le système de diagnostic et sera qualifiée d'inconnue.

2.5.4.3. Formalisation

Nous considérons $S_{i,j,k,l}$ le symptôme numéro l pouvant être généré par la fonction de surveillance et révélant un dysfonctionnement ou une défaillance du composant décrit dans la section 2.5.1. A partir de la description structurelle, il est possible de définir une défaillance $D_{i,j,k,m}$, défaillance numéro m du composant $CR_{i,j,k}$ tel que : $D_{i,j,k,m} = \Omega^S - \phi$, avec Ω^S l'ensemble des parties de l'ensemble S y compris l'ensemble vide (ϕ) sachant que S désigne l'ensemble des symptômes pouvant être révélés par la fonction de surveillance sur $CR_{i,j,k}$. $D_{i,j,k,m} \bigcap D_{i,j,k,n}$ peut être vide ou non.

Cette connaissance est représentée par des enregistrements dans un fichier XML. La structure de ces enregistrements est décrite sous la forme du schéma XML de la figure 2.12.

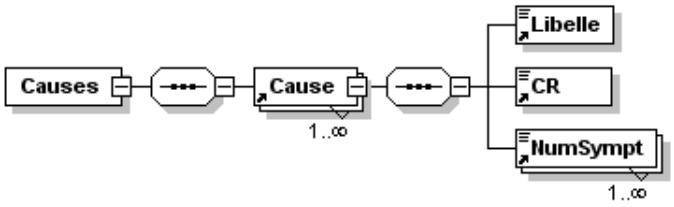


FIGURE 2.12: Schéma XML de la connaissance comportementale

2.5.5. Autres Connaissances

2.5.5.1. Les redondances

Lors de la description du système, il faut prendre en compte la présence de redondance de composants. Les redondances de composants sont mises en œuvre dans des systèmes comme les systèmes de transports (avions) ou les processus critiques (centrales nucléaires) afin d'augmenter la fiabilité d'une fonction qui, au lieu d'être assurée par un seul composant, l'est par plusieurs. Ils peuvent fonctionner simultanément, il s'agit alors de redondance active, ou lorsque le composant

assurant la fonction subit un dysfonctionnement un autre est alors activé et assure la fonction à la place du premier, il s'agit alors de redondance passive. La redondance peut être considérée comme une fonction à part entière. C'est cette considération que nous avons retenue dans la suite de ce mémoire. Le traitement particulier de son pronostic est décrit dans la section 2.4. Dans le cas du diagnostic, une fonction de redondance peut être considérée comme défaillante si plus de m composants parmi les n assurant la redondance sont défaillants sachant que $n-m$ est le nombre minimal de CR assurant la redondance permettant de conserver une fiabilité suffisante à la fonction. Une façon de représenter les redondances porte sur l'utilisation des arbres de défaillances.

2.5.5.2. L'arbre de défaillances pour l'aide au diagnostic

Les Arbres de Défaillances (AdD) permettent, à partir d'un ensemble de symptômes traduisant la défaillance d'une fonction, de descendre au niveau des composants dont le dysfonctionnement est cause de la défaillance (Limnios, 2005). Ainsi, en considérant la fonction implémentée par 3 composants $CR_{1,1,1}$, $CR_{1,1,2}$ et $CR_{1,1,3}$ présentée sur le schéma en haut à gauche de la figure 2.13, il est possible de construire l'AdD présenté sur cette même figure 2.13 où les boîtes représentent les symptômes, les portes logiques G02, G04 et G05 des portes logiques OU, la porte G03 une porte logique ET, B01, B02 et B03 des éléments de base (des symptômes) et T01 et T02 des éléments de base issus des résultats d'autres arbres de défaillances. T01 et T02 permettent de faire le lien avec la sortie d'autres arbres de défaillances afin de faciliter la lecture de modèles pouvant être très étendus.

L'exemple présenté permet de déterminer le composant défaillant lorsque la fonction ne génère plus de données. A partir de cet arbre, une coupe minimale peut être déterminée définissant ainsi les ensembles possibles d'événements de base (T0i, T02, B01, B02, B03) incriminant, dans ce cas, des composants qui peuvent conduire à l'événement redouté qui est qu'aucune donnée n'est fournie par le composant $CR_{1,1,3}$. Une coupe minimale représente la plus petite combinaison d'évènements pouvant conduire à l'événement indésirable ou redouté. On parle parfois également de « chemin critique ». La traduction de l'arbre de défaillances en porte logique ET uniquement permettrait alors de définir une défaillance fonctionnelle en fonction d'un ensemble de combinaisons de symptômes possible. Cela revient donc à définir de nouvelles entrées pour ce qui a été défini comme de la connaissance comportementale dont la modélisation est présentée dans la section 2.5.4.

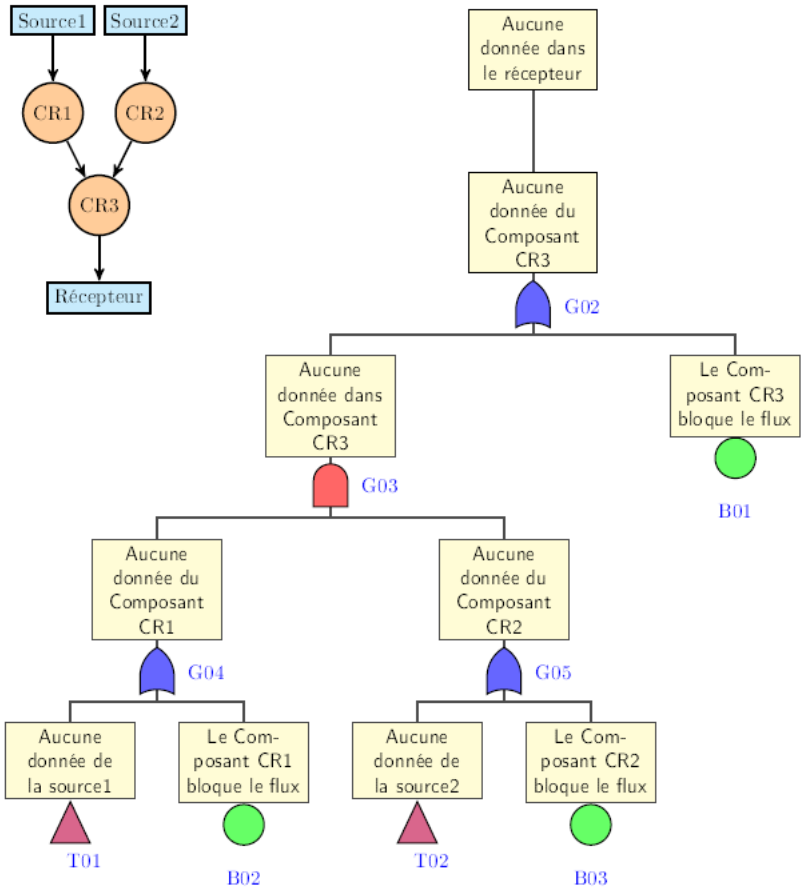


FIGURE 2.13: Arbre de défaillance dune fonction à 3 composants

A la vue des différentes connaissances utiles à l'élaboration du diagnostic et du pronostic d'un STCGD, il est donc possible de définir le contenu des connaissances $SKDiag$ et $SKProg$. Ainsi, $SKDiag = \{MS, MC, MT, MF\}$ et $SKProg = \{MS, MT, MF\}$. Pour les fonctions $Diag2$ et $Prog2$, $SKDiag$ et $SKProg$ peuvent se réduire à MF car c'est la seule qui permet de remonter d'un niveau composant à un niveau fonctionnalité.

2.6. Statuts des composants

Le fait qu'un composant fonctionne normalement et puisse être l'objet de différentes défaillances ou dysfonctionnements dont certains peuvent ne pas avoir été considérés lors de la phase de conception nécessite de compléter la déclaration de défaillance d'un composant par la fonction de diagnostic analysée section 2.3. Ceci nous conduit à définir quatre statuts pour les composants dont trois correspondent

à $AB(CR)$. Ces statuts peuvent être considérés comme des états qui doivent être identifiés par la fonction de diagnostic.

Le premier statut, que nous notons « OK », correspond au fonctionnement normal du composant et ceci quel que soit son mode de fonctionnement normal. Par exemple, une plateforme de calcul peut être, éteinte, en veille, en phase d'initialisation, en phase de configuration, en phase d'exécution des applications, en mise en veille, en sortie de veille... et ceci tant que les passages d'un de ces états à un autre font partie du fonctionnement normal de la plateforme. Le statut « OK » ne correspond évidemment pas à $AB(CR)$.

Le deuxième statut, que nous notons « DC » (Défaillance Connue), correspond à une défaillance connue du composant. C'est-à-dire que cette défaillance a été considérée lors de la phase de conception. La considération de cette défaillance a conduit les concepteurs à définir le ou les symptômes (natures et éventuellement valeurs) correspondant qui sont alors émis par la fonction de surveillance. La nature identifiée de la défaillance complète alors la déclaration de défaillance du diagnostic.

Le troisième statut, que nous notons « HS » (Hors Service), correspond à un comportement anormal d'un composant consécutif à une défaillance d'un autre composant dont il dépend structurellement. En reprenant l'exemple de la plateforme de calcul, celle-ci est éteinte alors qu'elle devrait être en phase d'exécution des applications selon le fonctionnement normal. Cependant, la source d'alimentation électrique qui lui fournit son énergie est défaillante. Elle ne lui fournit plus d'électricité et donc la plateforme est éteinte.

Enfin le quatrième statut, que nous notons « DI » (Défaillance Inconnue), lorsque le ou les symptômes transmis par la fonction de surveillance ne permettent pas de définir la nature de la défaillance du composant. Il peut notamment s'agir d'une défaillance qui n'a pas été considérée lors de la phase de conception.

Un composant ne peut avoir que l'un de ces quatre statuts et, comme cela est montré sur la figure 2.14, toute sortie de l'état « OK » conduit vers l'un des 3 autres états sans pouvoir en sortir. Cette considération implique que les composants du STCGD ne sont pas auto-réparables.

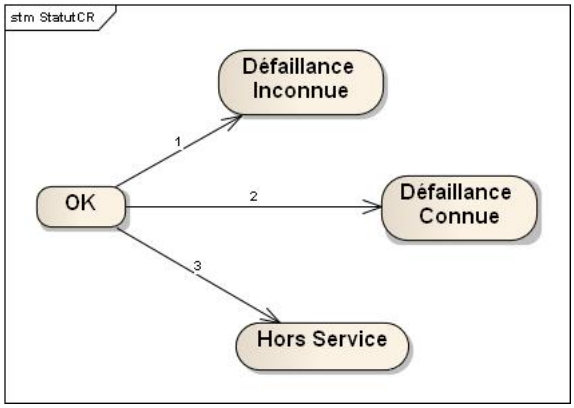


FIGURE 2.14: Statut d'un CR

2.7. Conclusion

L'analyse des fonctions de diagnostic et de pronostic a permis d'identifier les entrées de ces fonctions (symptômes pour le diagnostic, RUL pour le pronostic) fournies par la couche de surveillance, leurs sorties (défaillances avec statuts des composants et défaillances des fonctions pour le diagnostic, $TBAB$ et RUL des composants et $TBAB$ des fonctions pour le pronostic) et les connaissances nécessaires à leur mise œuvre. Nous avons donné des représentations de ces connaissances afin de les rendre exploitables par les traitements des fonctions de diagnostic et de pronostic. Le chapitre 3 est dédié à la présentation de principes pour la réalisation des fonctions de diagnostic et de pronostic contribuant à l'évaluation de l'état de santé des STCGD ainsi qu'à leur déploiement.

Chapitre 3

Evaluation de l'état de santé de systèmes techniques complexes de grande dimension

3.1. Introduction

La connaissance nécessaire à l'évaluation et au management de l'état de santé des STCGD a été présentée dans le précédent chapitre. Les connaissances structurelles et topologiques permettent l'affinage et la propagation des défaillances à travers le système. La connaissance comportementale permet à l'algorithme de diagnostic d'identifier le mode de fonctionnement des composants défaillants alors que la connaissance fonctionnelle permet aux fonctions de diagnostic et de pronostic de définir le statut ou le TBAB des fonctions connaissant celui des composants.

Nous avons vu qu'OSA-CBM définit une couche HA (Health Assessment) dont la fonction principale est de déterminer l'état de santé d'un système, sous-système, équipement ou composant surveillé en termes de faute, de défaillance et de disponibilité. Le module HA doit prendre en compte les diagnostics, les tendances issues de l'historique de santé, la charge, le statut opérationnel et l'historique de maintenance. La couche HA fournit l'état de santé (Health Status - HS) d'une entité surveillée. Dans (Vachtsevanos *et al.*, 2006), le pronostic et la gestion de l'état de santé (Prognostics and Health Management PHM) ont été définis comme une phase impliquant une prédiction du futur comportement du système en fonction de son état courant et des actions de maintenance à prévoir. Selon Mathur *et al.* (2001), le diagnostic et le pronostic sont des processus de management de l'état de santé d'un système. Les sept couches de l'architecture OSA-CBM sont rappelées sur la figure 3.1.

Le diagnostic est une estimation de l'état de santé actuel et passé du système basée sur des observations et le pronostic, une estimation de l'état de santé futur. Les différents résultats fournis par les fonctions de diagnostic et de pronostic doivent être considérés comme une aide à la maintenance pour prendre les décisions adéquates pour maintenir le système comme cela est considéré dans la couche de

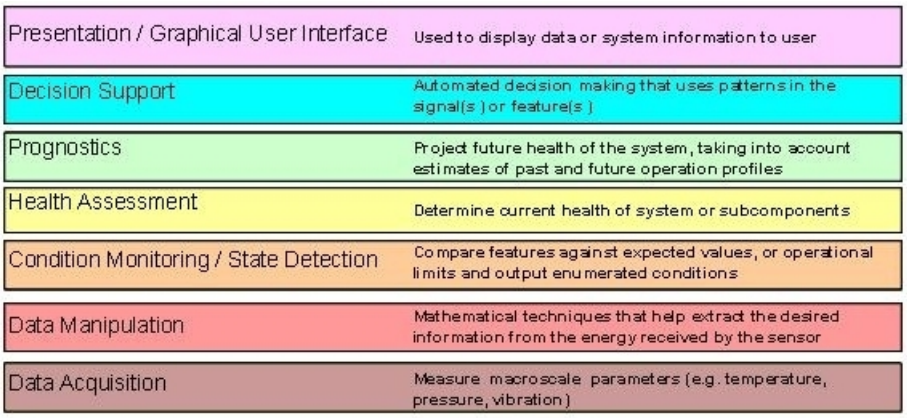


FIGURE 3.1: Les couches d'OSA-CBM

« Decision Support » d'OSA-CBM. Dans notre approche, le Health Management (HM) est introduit comme étant une combinaison des résultats du diagnostic et du pronostic. C'est pourquoi nous nous intéressons dans ce mémoire principalement aux couches de diagnostic et de pronostic. La figure 3.2 montre le diagramme de cas d'utilisation d'un système de HM d'un STCGD. Ce diagramme présente les utilisateurs potentiels d'un système d'évaluation d'état de santé d'un système ainsi que les différentes fonctions auxquelles il doit répondre. Ainsi, l'évaluation de l'état de santé d'un système passe par la définition de l'état de santé actuel, produit par la fonction diagnostic, et de l'état de santé futur, produit par la fonction pronostic, du système. Afin d'apporter un avis au niveau système, il faut, aussi bien pour les fonctions de diagnostic ou de pronostic, passer par une évaluation au niveau des fonctions du système, pour lesquelles il va falloir établir un résultat au niveau des composants implémentant ces fonctions.

L'évaluation de l'état de santé d'un STCGD passe par une évaluation de l'état de santé actuel du système, qui est fournit par la fonction de diagnostic, et une évaluation de la santé future du système qui est fournit par la fonction de pronostic. Les fonctions vont se baser sur les connaissances décrites à la section 2.5. Ce chapitre du mémoire vise à décrire les prérequis nécessaires pour la couche de détection pour la surveillance des STCGD. Un principe de fonctionnement des fonctions de diagnostic et de pronostic est ensuite proposé sous la forme de différents algorithmes. Enfin, une méthode d'évaluation de l'état de santé des STCGD est proposée.

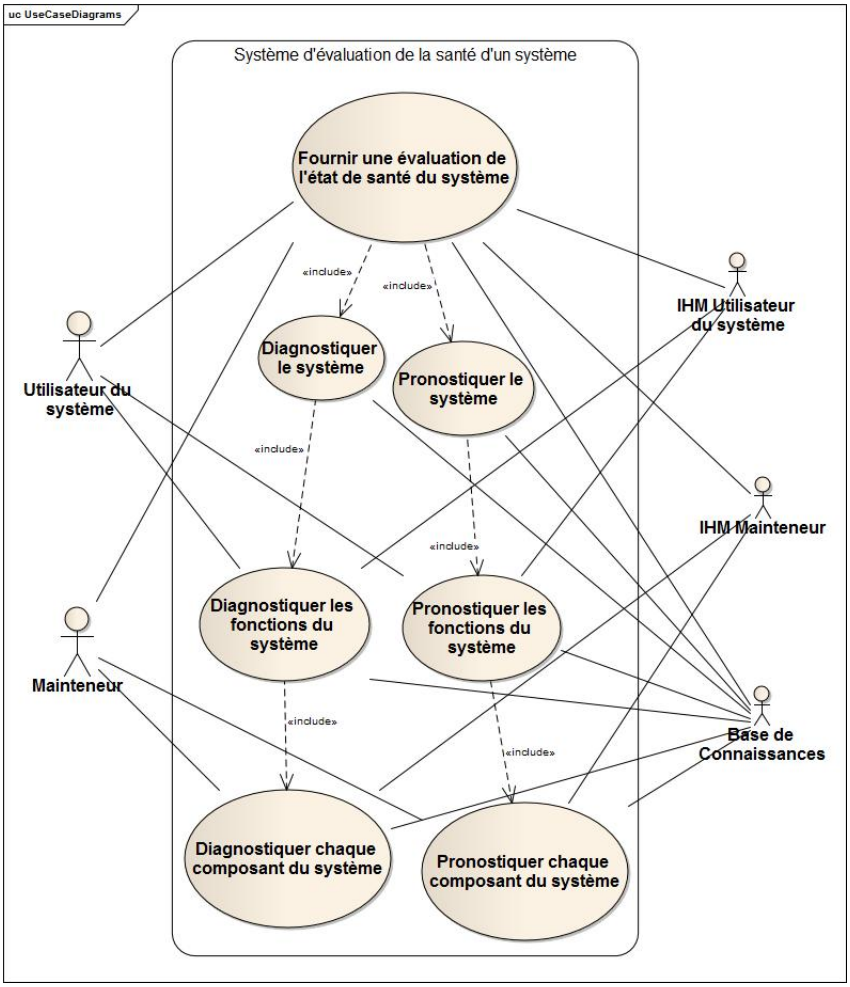


FIGURE 3.2: Diagramme de cas d'utilisation d'un système d'évaluation de la santé d'un système.

3.2. Pré-requis pour la surveillance d'un STCGD

Quelques pré-requis sont cependant nécessaires à la mise en œuvre des fonctions de diagnostic et de pronostic. Ils concernent plus particulièrement la couche de surveillance qui, comme nous l'avons précisé aux sections 2.3 et 2.4, doit fournir les symptômes à la fonction de diagnostic ainsi que les estimations de durée de vie résiduelle (RUL) à la fonction de pronostic.

3.2.1. Pré-requis pour les symptômes et contraintes des STCGD

Pour un STCGD, l'une des difficultés est d'appréhender le comportement du système dans sa totalité lorsqu'un de ses composants devient défaillant. En effet, le modèle et le comportement de chacun des composants constituant le système n'est parfaitement connu que du sous-traitant qui le fournit. De plus, pour surveiller ces systèmes, la couche de détection est en fait assurée par un ensemble de composants

logiciels. Un logiciel de surveillance peut avoir pour charge de détecter toute erreur sur un ou plusieurs composants à la fois. Pour assurer la surveillance de plusieurs composants à la fois, l'algorithme de surveillance doit réserver des fenêtres de détection pour chacun des composants qu'il doit surveiller. Cette période de surveillance est très souvent différente pour chacun des algorithmes de détection, ce qui conduit à des périodes d'échantillonnage différentes pouvant entraîner un problème lors de la reconstitution de la causalité dans l'ordre d'envoi des symptômes à la fonction de diagnostic par la couche de surveillance. Il s'avère alors qu'une fonction de diagnostic peut recevoir un symptôme et diagnostiquer un composant se comportant anormalement à cause, en fait, de la défaillance d'un autre composant dont le symptôme correspondant n'a pas encore été reçu par la fonction de diagnostic. Dans la structuration que nous proposons, la couche de surveillance met en œuvre les algorithmes de détection des comportements anormaux des composants. Ils exploitent pour cela des indicateurs. Un indicateur est une valeur évoluant dans le temps qui peut être simple, i.e. issue d'une mesure, ou complexe, i.e. élaborée à partir de plusieurs mesures et éventuellement de modèles. Un indicateur est défini et construit dans le but de révéler avec précision et certitude un défaut, une faute ou un type de défaillance d'un composant. La construction des indicateurs correspond souvent à des comparaisons du comportement actuel du composant surveillé, à des modèles de bon ou mauvais fonctionnement qui peuvent être parfois de simples intervalles de valeurs. Les indicateurs peuvent aussi être relatifs à l'état d'un composant ou à ses changements d'états. Tout écart entre un indicateur et ce qui en est attendu dans une situation donnée conduit la couche de surveillance à produire un symptôme qui est envoyé à la fonction de diagnostic. La fonction de diagnostic exploite les symptômes générés par la fonction de surveillance. La génération des symptômes doit être fiable, c'est-à-dire présentant de faibles probabilités de non détection et de fausse détection, pour que le diagnostic le soit aussi.

Dans le domaine industriel, les STCGD sont définis par des systémiers faisant appel à des sous-traitants ou fournisseurs. Une préconisation évidente serait d'exiger des sous-traitants de fournir la fonction de surveillance produisant les symptômes des équipements qu'ils fournissent ainsi que les relations entre ces symptômes et la nature des défaillances des équipements, du fait de leur parfaite connaissance de l'équipement. Une autre raison peut être qu'un suivi des symptômes générés ayant entrainé des interventions inutiles de sous-traitants ou de fournisseurs chez les clients du systémier, peut le conduire à faire engager la responsabilité de ses sous-traitants et de ses fournisseurs.

Toutefois, il est nécessaire de produire une spécification du format des symptômes destinés à être envoyés à la fonction diagnostic. Cette spécification doit au moins comprendre un identifiant et une date à laquelle ce symptôme a été généré. Cette date peut être extraite, notamment dans le cas d'utilisation de capteurs intelligents, de la datation des mesures à partir desquelles le symptôme a été déterminé.

3.2.2. Pré-requis pour les RUL

La couche de surveillance met en œuvre les algorithmes chargés de l'estimation des durées de vie résiduelles (RUL) des composants comme nous l'avons précisé dans la section 2.4. Le RUL d'un composant peut être déterminé par l'extrapolation de la tendance d'évolution d'un indicateur significatif et relatif à une dégradation conduisant à un état de défaillance du composant. Cette tendance est comparée à une limite prédéfinie à ne pas dépasser (Vachtsevanos *et al.*, 2006). Cependant, les RUL peuvent être issus d'autres indicateurs que ceux traduisant une dégradation comme le MTTF ou le MTBF définis à la section 1.3.3. Tout comme la génération de symptôme, le processus de génération de RUL est un processus permanent qui surveille le système tout au long de son utilisation.

La fonction de pronostic exploite les RUL générés par les algorithmes de la couche de surveillance. Pour permettre à la fonction de pronostic la production d'un résultat fiable, les RUL élaborés par les algorithmes de la couche de surveillance doivent l'être aussi. Cependant, il est difficile de déterminer un RUL avec précision. C'est pourquoi il est très souvent associé à une probabilité d'erreur. D'une façon générale, les systèmes complexes sont mis en œuvre par des systémiers faisant appel à des sous-traitants et fournisseurs. Une préconisation pourrait être de demander aux sous-traitants et fournisseurs de mettre en œuvre des fonctions générant les RUL de leurs équipements car ils connaissent parfaitement ce qu'ils implémentent. Le systémier n'a alors plus qu'à exploiter les RUL fournis dans la fonction de pronostic du STCGD. Pour cela, il est nécessaire de spécifier un format pour les RUL fournis par la couche de surveillance à la fonction de pronostic. Ce format doit comporter au moins un identifiant correspondant au composant auquel le RUL est relatif ou à une nature de défaillance du composant auquel le RUL est relatif. Dans ce cas le RUL du composant correspond au minimum des RUL associés aux différentes natures de défaillances du composant. Dans la suite de ce mémoire, nous ne considérons qu'un RUL par composant. Ce format doit, bien sûr, comporter la valeur de ce RUL dans une unité de temps préétablie et la date à laquelle ce RUL a été élaboré par la couche de surveillance.

A partir des RUL générés par la couche de surveillance, la fonction de pronostic établit la durée de bon fonctionnement restant pour chacun des composants ; ce qui correspond à la notion de TBAB défini à la section 2.4. Ce TBAB peut correspondre au RUL du composant ou au minimum des TBAB des composants dont il dépend pour fonctionner normalement.

3.3. Principe de diagnostic d'un STCGD

La réception, potentiellement désordonnée, des symptômes implique que l'une des premières activités que doit donc assurer la fonction de diagnostic est de remettre les données reçues dans leur ordre d'apparition et de permettre ainsi de faire apparaître les liens de causalité éventuels. L'algorithme de diagnostic modifie le statut des composants qu'il surveille. Ces statuts de composants sont conservés dans une mémoire qui peut être un fichier XML. A chaque fois qu'une information est reçue, le statut du composant incriminé et celui de la ou des fonctions auxquelles il contribue sont réévalués. A la réception d'un nouveau symptôme, la fonction de diagnostic doit l'insérer dans la liste contenant l'ensemble des symptômes reçus incriminant le même composant. La fonction maintient cette liste dans l'ordre chronologique d'apparition des symptômes. La fonction de diagnostic recherche ensuite le statut du composant avant la réception du nouveau symptôme et met à jour ce statut. Ce nouveau statut est daté et associé au symptôme reçu. Par ce biais, il est possible de voir l'évolution du statut des composants en fonction des données traitées. Comme les symptômes ne sont pas forcément reçus par la fonction de diagnostic dans l'ordre d'apparition, l'évolution du statut estimé d'un composant évolue comme présentée sur la figure 3.3 où lorsqu'un composant sort du statut « OK » il ne peut que demeurer dans le statut dans lequel il est passé. Cette réception potentiellement désordonnée des symptômes conduit la fonction de diagnostic à réviser son évaluation du statut d'un composant.

Les flèches rouges représentent les cas où l'ordre de réception des symptômes par la fonction de diagnostic ne correspond pas à l'ordre de leur apparition.

Les flèches numéro 1 et 2 correspondent à la situation où un symptôme incriminant un composant est reçu. Il y a alors deux situations : la première concerne le cas où le symptôme correspond à une situation de défaillance connue du composant dont l'état est alors déclaré comme « DC », la seconde concerne le cas où le symptôme incriminant le composant ne permet pas de statuer sur la nature de la défaillance du composant.

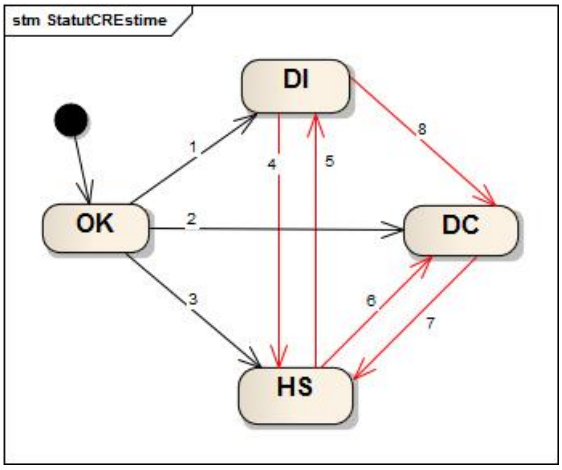


FIGURE 3.3: Statut d'un composant estimé par la fonction de diagnostic

La flèche numéro 3 correspond à la situation où un premier composant est déclaré dans le statut « HS » suite à au moins une modification du statut déclaré par un autre composant auquel il est structurellement (ou topologiquement) lié en entrée.

Les flèches numéro 5, dans le cas d'une défaillance inconnue, et 6 dans le cas d'une défaillance connue de la figure 3.3 correspondent à la situation où un premier composant a été déclaré hors service « HS » suite à la réception d'un ou plusieurs symptôme incriminant un second composant dont le premier composant est dépendant (structurellement ou topologiquement). La réception d'un nouveau symptôme incriminant le premier composant dont la date d'apparition est antérieure à ceux incriminant le second composant fait que le statut de ce premier passe à « DI » si le symptôme ne correspond pas à un état de défaillance connu ou à « DC » dans le cas contraire.

Les flèches numéro 4 et 7 de la figure 3.3 concernent le cas où la couche de surveillance a envoyé des symptômes incriminant un premier composant qui est dépendant (structurellement voire topologiquement) d'un second composant pour lequel la fonction de diagnostic vient de recevoir un symptôme dont la date d'apparition est antérieure à ceux reçus pour le premier composant. Dans cette situation, le premier composant a été déclaré dans le statut « DC » si le ou les symptômes préalablement reçus correspondaient à un état de défaillance préalablement connu ou dans le statut « DI » dans le cas contraire. La réception du symptôme incriminant le second composant possédant une date d'apparition antérieure à ceux reçus pour le premier composant conduit à considérer que la défaillance du premier composant correspond plutôt à un état hors service (« HS ») du fait de l'antériorité de la défaillance du second composant. Dans cette situation la déclaration en

statut « HS » du premier composant est privilégiée. Dans la pratique, cela conduit les agents à privilégier le remplacement ou la réparation du second composant. Cependant, il convient de vérifier le bon fonctionnement du premier composant une fois le remplacement ou la réparation du second effectué.

La flèche numéro 8 correspond à la situation pour laquelle un composant a été déclaré dans le statut « DI » car les symptômes l'incriminant reçus jusqu'alors ne correspondent pas à un état de défaillance connu du composant. La réception d'un nouveau symptôme incriminant ce composant fait correspondre alors la liste des symptômes reçus à un état de défaillance connu « DC » et répertorié dans le modèle comportemental du composant. Dans la pratique, cela revient à privilégier le statut « DC » plutôt que le statut « DI ». Quoi qu'il en soit, un composant présentant une évaluation de son état « DC » ou « DI » doit être remplacé ou subir une réparation. Les changements, décrit par les flèches 4 à 8 de la figure 3.3, de l'évaluation du statut d'un composant par la fonction de diagnostic dont nous venons de présenter le mécanisme implique un processus de diagnostic non-monotone. C'est-à-dire que l'évaluation de l'état d'un composant peut être remise en question à la réception de chaque nouveau symptôme. Cette non-monotonie du diagnostic est principalement due au fait que les symptômes ne sont pas forcément reçus dans l'ordre de leur apparition par la fonction de diagnostic. Ceci est notamment la conséquence du fait que, dans les STCGD, les fréquences d'échantillonnage des informations à partir desquelles les symptômes sont générés diffèrent, que la durée d'élaboration des symptômes par les algorithmes de surveillance n'est pas constante et que le temps d'acheminement des symptômes vers la fonction de diagnostic peut être variable. Les symptômes doivent donc être datés par les algorithmes de la couche de surveillance. La notion de session doit alors être introduite pour la fonction de diagnostic d'un STCGD telle que nous la proposons. En effet, la fonction de diagnostic produit alors, parmi d'autres sorties, une évaluation du statut d'un composant en fonction des symptômes qu'elle a reçu à la date courante. Le diagnostic, en termes de statut des composants, qui a la plus grande probabilité de correspondre aux statuts réels des composants est celui qui précède l'instant auquel la fonction de diagnostic est arrêtée. En effet, tant que la fonction de diagnostic n'est pas arrêtée, cela veut dire qu'elle n'a toujours pas convergé vers une solution. D'autres symptômes peuvent être reçus et auront pour conséquence de confirmer ou d'infirmer le diagnostic alors établi. La fonction de diagnostic définit un ensemble de composants défaillants (processus d'isolation) et définit les causes (processus d'identification). La défaillance d'un composant est ensuite propagée dans le système pour affiner le diagnostic et déclarer les états

« HS » des composants ne pouvant consécutivement plus fonctionner normalement. Lorsque le statut d'un ou de plusieurs composants change, la fonction de diagnostic évalue l'état fonctionnel correspondant à la déclaration de défaillance de fonctions. La cause de la défaillance fonctionnelle est traduite en termes de composants défaillants. Les processus d'isolation et d'identification exploitent la connaissance comportementale décrite dans la section 2.5.4. Le processus de propagation qui évalue les répercutions sur le fonctionnement d'autres composants exploite les connaissances structurelles et topologiques également décrites dans les sections 2.5.1 et 2.5.3. Cette première étape de la fonction diagnostic correspond à la fonction Diag1 décrite dans la section 2.3. Le processus qui évalue les répercutions de la défaillance de composants sur les fonctions du STCGD exploite la connaissance fonctionnelle décrite dans la section 2.5.2. Cette seconde étape de la fonction diagnostic correspond à la fonction Diag2 décrite dans la section 2.3.

Les fonctions Diag1 et Diag2 requièrent éventuellement des tests. Nous considérons dans la suite de ce mémoire que ces tests peuvent être omis car leur objectifs est de fournir d'éventuels symptômes supplémentaires permettant d'améliorer l'évaluation de l'état de santé des composants, des fonctions et donc du STCGD. Ces tests peuvent donc être assimilés à des symptômes supplémentaires.

La fonction Diag1 fonctionne selon le principe décrit par l'algorithme 3.1 suivant après réception d'un symptôme :

Algorithm 3.1 Algorithme de diagnostic : Diag1

```
Entrées : symptôme S, connaissances MS, MT, MC
Entrée-Sortie : Liste des évaluations des statuts des composants L.
Variables :
Composant CRI
Listes des symptômes d'un composant LScri
Statut d'un composant AbCRI,
AbCRIold //comporte l'état et la nature de la défaillance s'il y a
Début
  CRI ← Recherche du composant incriminé (S)
  LScri ← Insertion du symptôme dans la liste ordonnée des symptômes du
  composant (S, CRI, LScri)
  AbCRIold ← AbCRI
  AbCRI ← Evaluation de la nature de la défaillance du composant incriminé
  (MC, LScri, AbCRI)
  L ← Mise à jour (AbCRI, L)
  Si AbCRIold.état ≠ AbCRI.état
    L ← Propagation (AbCRI, L, MS, MT)
  Fin si
Fin
```

Les fonctions Evaluation de la nature de la défaillance du composant incriminé et Propagation fonctionnent selon les principes schématisés sur la figure 3.3. La fonction Diag2, quant à elle, est décrite par l'algorithme 3.2 suivant :

Algorithm 3.2 Algorithme de diagnostic : Diag2

Entrées :
Liste des évaluations des statuts des composants LC
Connaissances MF
Entrée-Sortie : Liste des évaluations des statuts des fonctions LF
Début
  LF ← Mise à jour des statuts des fonctions (LF, MF, LC)
Fin

L'algorithme 3.2 de Diag2 consiste donc uniquement à mettre à jour les statuts des fonctions en fonction de ceux des composants de la façon décrite dans cette section. Cette mise à jour n'est nécessaire que si une évaluation de statut d'un composant a changé. Les entrées-sorties formant les listes LC et LF de ces fonctions correspondent aux ensembles L1 et L2 de la section 2.3.

La figure 3.4 présente, sous la forme d'un diagramme de temps, pour le scénario A considéré l'évolution du résultat de diagnostic estimé par l'algorithme 3.1 pour les CR et par l'algorithme 3.2 pour le diagnostic de la fonction implémentée par les CR. Dans le scénario de défaillances, une fonction $F_{1,1}$ composée de 3 composants notés $CR_{1,1,1}$ à $CR_{1,1,3}$ structurellement liés est considérée. $CR_{1,1,3}$ est structurellement lié à $CR_{1,1,1}$. Le scénario est le suivant : apparition du symptôme $S_{1,1,2,1}$ (symptôme numéro 1 incriminant le $CR_{1,1,2}$) puis $S_{1,1,3,2}$ (symptôme numéro 2 du $CR_{1,1,3}$).

Du fait de la dépendance structurelle et de l'apparition d'un symptôme incriminant le composant $CR_{1,1,3}$, un autre symptôme $S_{1,1,1,1}$ est généré incriminant le $CR_{1,1,1}$. La fréquence d'envoi et de détection de la couche surveillance du $CR_{1,1,1}$ est plus rapide que celle du $CR_{1,1,3}$. L'agent de diagnostic surveillant $F_{1,1}$ implémentée par les 3 composants reçoit le symptôme $S_{1,1,1,1}$ avant $S_{1,1,3,2}$. De plus la défaillance du $CR_{1,1,2}$ est connue à la réception de $S_{1,1,2,1}$. Le diagnostic attendu est $CR_{1,1,1}$: « HS » car il y a une défaillance du $CR_{1,1,3}$, $CR_{1,1,2}$: « DC » plus la nature de la défaillance qui est ignorée dans ce cas et $CR_{1,1,3}$: « DI ». Pour $F_{1,1}$, le diagnostic est une défaillance connue ayant pour origine le $CR_{1,1,2}$.

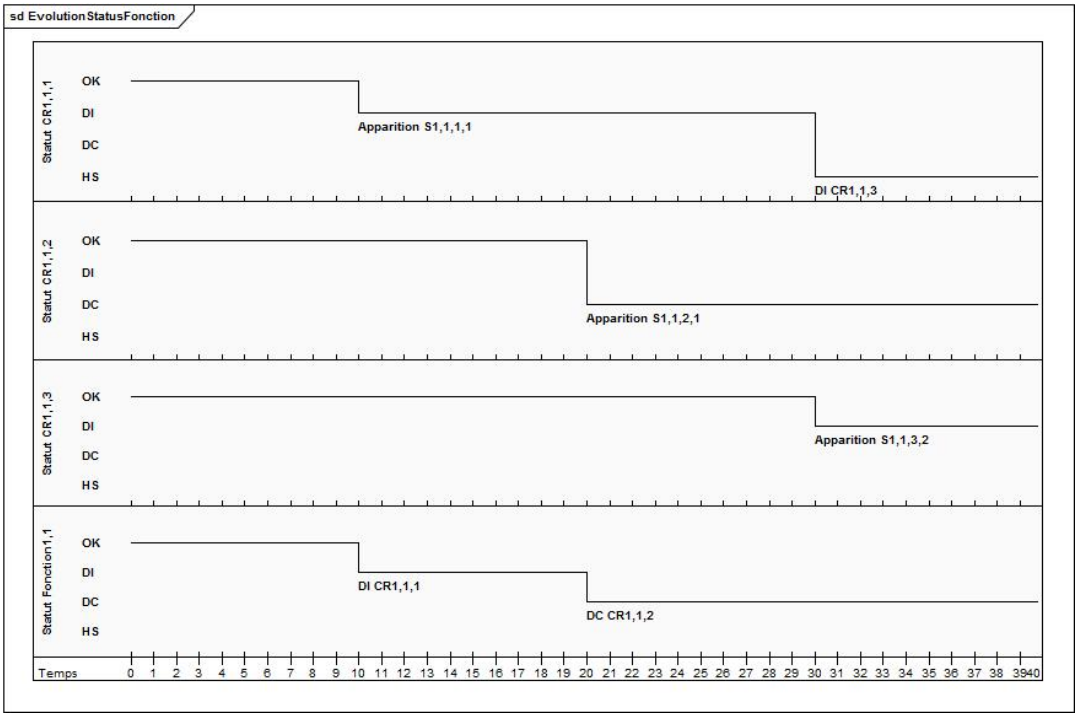


FIGURE 3.4: Evolution du statut estimé des composants et des fonctions incriminés pour le scénario A

Finalement, nous avons mis en évidence l'efficacité de l'algorithme de diagnostic pour évaluer et réévaluer le statut des composants à diagnostiquer en se basant pour cela sur les connaissances disponibles et les dates d'apparition des symptômes générés. La finalité de la fonction de diagnostic est d'évaluer le statut de chaque composant du système en fonction des symptômes reçus de la couche de surveillance et des connaissances comportementales, structurelles et topologiques. A partir des évaluations du statut de chaque composant du système et des connaissances fonctionnelles une évaluation de l'état des fonctions est effectuée. En effet, après la réception d'un symptôme, la fonction Diag1 est exécutée puis, si le statut d'un composant a changé, la fonction Diag2 est appelée.

Cependant, ces évaluations d'états de composants ne sont pas suffisantes au regard de la maintenance préventive conditionnelle car elles ne permettent que le constat de défaillances ne pouvant entraîner que des interventions de maintenance curative. L'implémentation d'une fonction de pronostic contribue, par les résultats qu'elle fournit, à compléter l'évaluation de la santé du STCGD en donnant une image de ces capacités opérationnelles futures autorisant ainsi la planification d'interventions de maintenance préventive.

3.4. Principe de pronostic d'un STCGD

L'état de santé d'un STCGD composé d'éléments de diagnostic sur les composants et les fonctions du système est complété par des données issues d'une fonction de pronostic. Comme nous l'avons décrit dans la section 2.4, la fonction de pronostic exploite les durées de vie résiduelles (RUL) des composants fournis par la couche de surveillance. A partir des RUL des composants, les durées résiduelles de bon fonctionnement (TBAB) des composants sont évaluées de la façon décrite à la section 2.4. Cette étape est mise en œuvre par la fonction Prog1. La fonction Prog2, décrite également à la section 2.4, a pour but d'évaluer le TBAB des fonctions à partir des TBAB des composants déterminés par la fonction Prog1.

L'algorithme de pronostic utilise lui aussi les connaissances structurelles et topologiques pour propager les résultats de pronostic et génère alors le TBAB du ou des composants en suivant un algorithme présenté par la suite. La notion de TBAB a été introduite car la notion de RUL correspond à une donnée qui est générée uniquement par la couche monitoring. Cependant, lors d'une propagation de RUL, il faudra la remplacer par celle de TBAB après cette explication. Il se peut que le RUL du composant en aval de la dépendance soit supérieur au RUL qui lui a été propagé i.e. le RUL du composant en amont. Le RUL du composant deviendrait celui nouvellement reçu. Cependant, si le RUL qui a été envoyé est remis à jour et devient supérieur au RUL du composant recevant de nouveau l'information, cette fois-ci, le RUL du composant sera égal à son propre RUL , c'est-à-dire celui envoyé par la couche de surveillance. Cependant, cette information est perdue car écrasée par l'ancienne valeur. Il fallait donc recourir à un autre concept pour définir cette situation où le comportement du composant deviendra anormal, soit de lui-même, soit par le biais d'un composant devenant défaillant avant ce dernier et dépendant de celui-ci. Un composant se situant en amont d'une chaîne structurelle verra son TBAB toujours égal à son RUL. Si le RUL d'un composant, émis par la couche de surveillance, est associé à la nature de défaillance de ce même composant, les connaissances topologiques doivent être utilisées afin d'évaluer le TBAB des composants situés à proximité.

La fonction Prog est décrite par l'algorithme 3.3 lorsque qu'un RUL est reçu de la couche de surveillance.

Les fonctions de propagation correspondent aux principes d'évaluation des TBAB des composants et des fonctions décrites à la section 2.4. Les entrées-sorties formant les listes LRT et LFT de ces fonctions correspondent aux ensembles H1 et H2 de la section 2.4.

Algorithm 3.3 Algorithme de pronostic : Prog

```
Entrée :
RUL d'un composant R(C) // comporte la nature de la défaillance à venir
Connaissances MS, MT, MF
Entrées-Sorties :
Liste des RUL des composants LRC
Liste des TBAB des composants LRT
Liste des TBAB des fonctions LFT
Variables :
TBAB d'un composant TBAB(C)
Début
  LRC ← Mise à jour (LRC, R(C))
  Si R(C).durée < TBAB(C).durée
    TBAB(C) ← R(C)
    LRT ← Propagation (LRT, TBAB(C), MS, MT)
    LFT ← Propagation (LFT, LRT, MF) //correspond à la fonction Prog2
  Fin si
Fin
```

L'association des résultats produits par la fonction de pronostic aux résultats produits par la fonction de diagnostic permet l'évaluation de l'état de santé d'un STCGD.

3.5. Méthode d'évaluation de l'état de santé d'un STCGD

L'état de santé du système est produit par les fonctions de diagnostic et de pronostic. Les données élaborées par ces fonctions, listes de RUL, de TBAB, de CR en faute et de fonctions défaillantes sont différemment interprétées en fonction de l'utilisateur, i.e. opérateur de maintenance ou opérateur sur système. L'évaluation de l'état de santé d'un composant ou d'une fonction X, notée $HS(X)$ peut être définie par :

$$HS(X) = (St(X), TBAB(X))$$

où $St(X)$ représente le statut d'un composant ou d'une fonction X.

Le HS du système complexe (STCGD) est défini par :

$$HS(STCGD) = (\Delta 2, \Pi 2)$$

où $\Delta 2$ représente l'ensemble des St de toutes les fonctions du système et $\Pi 2$, l'ensemble des TBAB de toutes les fonctions du système.

3.6. Conclusion

Les fonctions de diagnostic et de pronostic décrites permettent d'évaluer l'état de santé des STCGD. Les algorithmes décrivant les principes proposés de diagnostic et de pronostic exploitent les types de connaissances décrits à la section 2.5 pour affiner leurs résultats. La prise en compte du temps d'apparition des données dans le processus de diagnostic permet de révéler un lien de causalité dans l'ordre d'apparition des événements. Ceci permet alors de reconsidérer le statut des composants déclarés comme défaillant ou hors service suite à la réception de symptômes dans un ordre ne respectant pas nécessairement la chronologie due à ce lien de causalité. Ces reconsidérations nécessaires dans les STCGD requièrent donc un principe de diagnostic non monotone que nous avons proposé dans ce chapitre. Le principe de pronostic basé sur la notion de temps restant avant un comportement anormal (TBAB) et proposé dans ce chapitre concourt avec le principe de diagnostic à l'évaluation de l'état de santé d'un STCGD par l'association de leurs résultats respectifs. Cependant, ces principes doivent être déployés. Des déploiements sont proposés dans le chapitre suivant. Ils y font l'objet de comparaisons de performances obtenues à partir d'une plateforme spécifiquement développée pour permettre de simuler des benchmarks de STCGD.

Chapitre 4

Plateforme d'évaluation de la santé de systèmes techniques complexes de grande dimension

4.1. Introduction

L'évaluation de l'état de santé d'un système nécessite la mise en œuvre de fonctions de diagnostic et de pronostic dont les principes ont été décrits dans le chapitre précédent et exploitent les différentes connaissances disponibles sur le fonctionnement du système présentées dans le chapitre 2. L'association des résultats produits par ces deux fonctions fournit une évaluation de l'état de santé des composants et fonctions du système. Cependant, les principes des fonctions requièrent un déploiement pour être évalués en termes de performances. Une architecture générique permettant le déploiement de fonctions de diagnostic et de pronostic a été développée. Plusieurs déploiements des fonctions de diagnostic et de pronostic sont étudiés. Une proposition de déploiement distribué sera discutée. Celui-ci a fait l'objet d'une comparaison de performances avec un déploiement centralisé. Chacun des déploiements a été étudié de façon approfondie et leurs fonctionnements sont présentés. Les données échangées entre les différents agents de l'architecture sont caractérisées. Les différents modes de transmission des données sont présentés et discutés. Afin d'effectuer la comparaison entre les différents déploiements possible, une plateforme a été développée. Son objectif est de simuler la couche de surveillance d'un STCGD. Un cas d'étude est proposé pour chacun des deux déploiements et leurs performances sont comparées. Pour se faire, les vitesses de convergence des deux déploiements ont été évaluées et confrontées. Cela nous a conduit à modifier le cas d'étude, et plus précisément le générateur de symptômes pour surcharger artificiellement le benchmark pour mettre en avant l'apport du distribué dans ce cas de figure. L'évaluation de l'équilibrage de la charge des agents de diagnostic constituant l'architecture distribuée a permis de relever l'importance d'avoir un bon équilibrage afin que l'apport du distribué au centralisé soit optimal en termes de vitesse de convergence.

4.2. Déploiement distribué du diagnostic et du pronostic d'un STCGD

4.2.1. Présentation de l'architecture

Les architectures distribuées permettent, comme cela a été présenté dans la section 2.5, d'améliorer la flexibilité d'un système par la possibilité d'ajouter ou de retirer facilement des éléments matériels ou logiciels, et donc, de proposer de nouveaux services ou d'améliorer ceux existants. Les mises à jour, mises à niveau et maintenances sont ainsi facilitées (Zennir, 2004). Les travaux menés sur la distribution de la fonction de diagnostic ont montré l'applicabilité de tels déploiements notamment sur des systèmes embarqués (Biteus *et al.*, 2008). Cependant, les distributions proposées sont des répartitions de sous-fonctions composant la fonction de diagnostic (notamment détection, isolation et identification) dont certaines sont, dans notre cas, des tâches dévolues à la couche de surveillance. Ces distributions consistent, pour certaines, à des implémentations de plusieurs techniques de diagnostic fonctionnant en parallèle, et la décision sur l'état du système est alors issue d'une combinatoire des résultats de ces fonctions (Touaf, 2005). L'architecture que nous proposons ne repose pas sur de telles approches. Il s'agit d'une distribution des principes de diagnostic et de pronostic exposés dans le chapitre 3, sur des parties du STCGD. Nous avons considéré que ces parties seraient les fonctions du STCGD. Cette distribution, fonction d'une « cartographie », ici, fonctionnelle du STCGD présente, notamment, comme avantage une certaine généricité des éléments composants l'architecture distribuée de l'évaluation de l'état de santé d'un STCGD. En effet, les éléments chargés du diagnostic mettent en œuvre le même principe de diagnostic mais sur des fonctions du STCGD différentes. Il en est de même pour les éléments mettant en œuvre le principe de pronostic. L'architecture proposée est représentée sur la figure 4.1. Elle est constituée de cinq types d'éléments :

- Un élément représentant la couche de surveillance émet les symptômes et les RUL. Cet élément, représenté ici de façon unique, peut être constitué de plusieurs moniteurs chargés d'émettre les éventuels symptômes et/ou RUL.
- Une Base de données (BdD) permet aux agents de l'architecture d'avoir l'accès aux connaissances structurelles, comportementales et topologiques nécessaires aux traitements qu'ils effectuent. Cette base de données, représentée ici de façon unique, pourrait être distribuée. Des exemples de distributions pourraient être une base par type de connaissance ou encore une base par fonction.
- Un élément appelé « blackboard » (tableau noir) et noté BB sur la figure

4.1, agit comme une mémoire partagée et permet de collecter et de rendre disponible à l'ensemble des agents de l'architecture les informations sur l'évaluation de la santé du système élaborées par eux-mêmes.

- Des Agents de Diagnostic (AD) élaborent le diagnostic des composants et des fonctions du STCGD selon le principe décrit à la section 3.3.
- Des Agents de Pronostic (AP) élaborent le pronostic des composants et des fonctions du STCGD selon le principe décrit à la section 3.4.

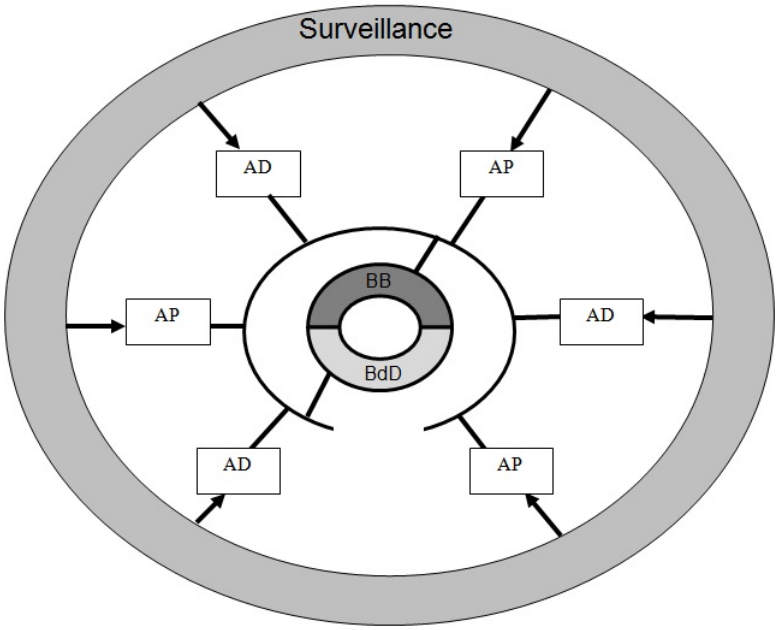


FIGURE 4.1: Déploiement distribué d'une fonction d'évaluation de l'état de santé d'un STCGD

Comme nous l'avons mentionné au début de ce paragraphe, il y a un agent de diagnostic et un agent de pronostic par fonction du STCGD. Ces éléments sont appelés « agent » car ils répondent à la définition de Jennings et Wooldridge (1995) donnée dans la section 1.4.1. En effet :

- ils conduisent la tâche pour laquelle ils sont conçus qui est le diagnostic ou le pronostic,
- ils interagissent avec leur environnement, notamment par l'échange de messages,
- ils disposent d'une forme de perception de leur environnement par les informations envoyées par la couche de surveillance et les messages envoyés par les autres agents,

- ils produisent des réponses dans un temps donné, cependant ce temps dépend de la durée du traitement nécessaire à la réalisation des principes de diagnostic ou de pronostic,
- ils prennent des initiatives, si la prise de décision, bien qu'automatique, quant à l'évaluation de l'état de santé de composants ou de fonctions peut être assimilée à une forme d'initiative.

Dans cette architecture, tous les agents de diagnostic ne reçoivent pas tous les symptômes. Seuls les symptômes concernant les composants implémentant la fonction diagnostiquée par l'agent sont reçus par lui. Il en est de même pour les agents de pronostic et les RUL. Toutefois, des liens structurels et/ou topologiques entre composants implémentant des fonctions différentes existent. L'existence de ces liens nécessite des échanges de messages entre les agents de diagnostic mais aussi entre les agents de pronostic. En effet, lorsqu'un agent de diagnostic évalue qu'un composant est défaillant, il doit en informer les agents en charge du diagnostic de fonctions implémentées par des composants qui en sont dépendants structurellement voire topologiquement. Les messages issus de cette situation sont appelés dans la suite de ce mémoire des « Faulty_LRU_Message ». Cette situation est également rencontrée dans le cas du pronostic où un agent ayant réévalué le TBAB d'un composant à la baisse doit signaler cette modification aux agents en charge du pronostic de fonctions implémentées par des composants qui en sont dépendants structurellement voire topologiquement. Les messages issus de cette situation sont appelés dans la suite de ce mémoire des « TBAB_Message ». Cette répartition sur plusieurs agents des traitements de diagnostic et de pronostic nécessitent donc des flux d'informations supplémentaires que sont les « Faulty_LRU_Message » et les « TBAB_Message ». Les agents de diagnostic et de pronostic doivent donc être également capables d'actualiser, à partir de ces flux, leur évaluation de l'état de santé des composants et de la fonction dont ils ont la charge. Afin d'assurer la communication entre les éléments de l'architecture, ceux-ci sont considérés comme des clients et/ou comme des serveurs. Dans l'approche client/serveur, un client envoie une requête à un serveur. Le serveur effectue le traitement correspondant à la requête et renvoie, les cas échéant, la réponse au client (Gardarin et Gardarin, 1996). Deux modes de communication peuvent être employés pour les échanges dans l'approche client/serveur. Il s'agit du mode synchrone et du mode asynchrone. Dans le mode synchrone, lorsqu'un client envoie une requête au serveur, le client attend que le serveur lui réponde avant de poursuivre son traitement. Ce mode de communication est décrit sur le schéma de la figure 4.2. Dans le mode asynchrone, lorsqu'un client envoie une requête à un serveur et le client peut poursuivre les

traitements qu'il effectue sans attendre la réponse du serveur. Le serveur et le client fonctionne alors en parallèle. Lorsque le serveur a terminé d'effectuer les traitements liés à la requête du client, il lui transmet la réponse. Le client récupère alors le résultat et poursuit son traitement. Ce mode de communication est décrit sur le schéma de la figure 4.3. Dans le cas où le serveur ne peut traiter qu'une seule donnée à la fois et qu'il reçoit une nouvelle requête alors qu'il est déjà en train d'en traiter une, celui-ci conserve la donnée envoyée dans une file d'attente qu'il traitera après avoir terminé le traitement de la précédente.

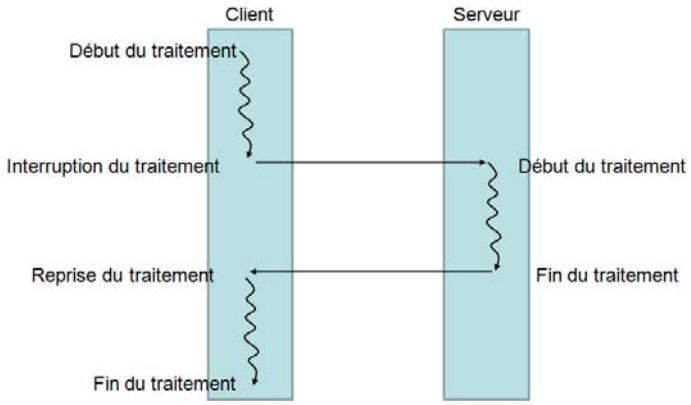


FIGURE 4.2: Mode de communication synchrone client/serveur

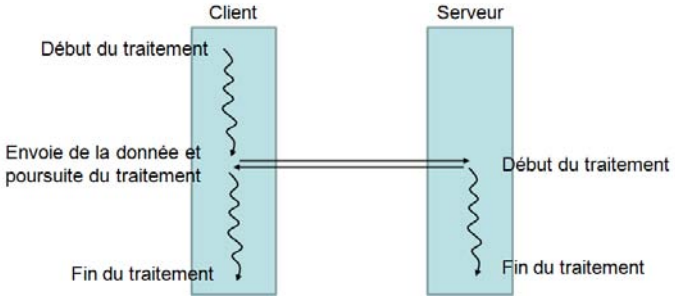


FIGURE 4.3: Mode de communication asynchrone client/serveur

Dans le cas des échanges entre la couche de surveillance (qui est dans ce cas un client) et les agents de diagnostic et de pronostic (qui sont alors des serveurs), la couche de surveillance n'a pas à attendre les résultats produits par les agents de diagnostic et de pronostic. En effet, les résultats produits par les agents de diagnostic et de pronostic que sont les AB() et les TBAB() qui constituent l'évaluation

de l'état de santé du STCGD ne sont pas destinés à la couche de surveillance mais à des applications d'aide à la décision pour la maintenance, ou la planification. Dans le cas des échanges entre la base de données et les agents de pronostic ou de diagnostic, la base de données ne fait que répondre à des requêtes émises par ces agents. Elle peut donc être considérée comme un serveur. Il en est de même pour les échanges entre le « blackboard » et les agents de pronostic ou de diagnostic. Il peut donc être considéré comme un serveur. Les agents, quant à eux, sont donc des serveurs pour la couche de surveillance et des clients pour le « blackboard » et la base de données. La distribution des principes de diagnostic et de pronostic qui est considérée dans cette section nécessite des précisions par rapport à la description de ces mêmes principes faite dans le chapitre 3. Ces précisions permettent, notamment, de tenir compte du mode de communication et de la réception par un agent de diagnostic d'un « Faulty_LRU_Message » et par un agent de pronostic d'un « TBAB_Message ».

Concernant le mode de communication nous pensons que le mode asynchrone est plus approprié entre la couche de surveillance et les agents de diagnostic et de pronostic. En effet, la couche de surveillance n'a ainsi pas besoin d'attendre la fin des traitements des agents pour poursuivre ses activités. Nous pensons que ce mode de communication est également approprié dans les échanges entre les agents. Ainsi, un agent poursuit son activité malgré la réception d'un message émis par un autre agent. Ce message est alors mis en file d'attente et traité lorsque les autres éléments de la file auront été traités. Toutefois, nous pensons que la communication entre les agents et la base de données ou le « blackboard » doit plutôt correspondre au mode synchrone. D'une part, parce que les agents ne peuvent pas poursuivre leurs traitements tant qu'ils n'ont pas reçu les informations demandées à la base de données et, d'autre part, parce que ce mode de communication évite les accès concurrents lors de l'écriture d'informations dans le « blackboard ». Il permet également aux agents d'obtenir les éléments sur les états de santé évalués des composants et fonctions.

4.2.2. Déploiement distribué du diagnostic

Dans le principe de distribution exposé dans la section précédente, le principe de diagnostic est distribué aux agents de diagnostic. Tous les agents de diagnostic effectuent le même traitement. Cependant, ils sont dédiés à des fonctions différentes du STCGD. Lorsqu'un agent de diagnostic reçoit un symptôme de la couche de surveillance, il agit conformément à l'algorithme 4.1. Ce processus, mené par un agent de diagnostic, peut également être décrit dans l'architecture à l'aide du

diagramme de séquence de la figure 4.4 représentant les activités des éléments de l'architecture ainsi que les échanges de messages.

Algorithm 4.1 Algorithme de traitement d'un symptôme par la fonction diagnostic

Entrées : symptôme, connaissances MC, MS, MF, MT
Entrée - Sortie : Liste des évaluations des statuts des composants et des fonctions
Début
  Rechercher quel est le composant incriminé par le symptôme,
  Insérer ce symptôme dans la liste chronologique des données reçus incriminant le composant,
  Evaluer la nature de la défaillance du composant à l'aide des connaissances comportementales,
  Mettre à jour l'évaluation de la défaillance du composant (statut «DI» ou «DC» accompagné de la nature de la défaillance),
  Si le statut du composant a changé à cause de ce symptôme,
    Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant la fonction qui sont structurellement ou topologiquement dépendant du composant.
    Changer le statut des composants ainsi dépendant selon la règle suivante :
      - passage du statut d'un composant ainsi dépendant à « HS » si son statut était « OK »
      - passage du statut d'un composant ainsi dépendant à « HS » si ce dernier symptôme reçu à une date d'apparition antérieure à toutes celles des symptômes du composant ainsi dépendant, que son ancien statut fut « DI » ou « DC »,
    Evaluer le statut de la fonction,
    Inscrire les statuts mis à jour dans le « blackboard »,
    Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant d'autres fonctions dont des composants sont dépendants des composants dont le statut a changé.
    Envoyer un « Faulty_LRU_Message » par composant des fonctions concernées aux agents en charge de leur diagnostic,
  Fin Si
Fin

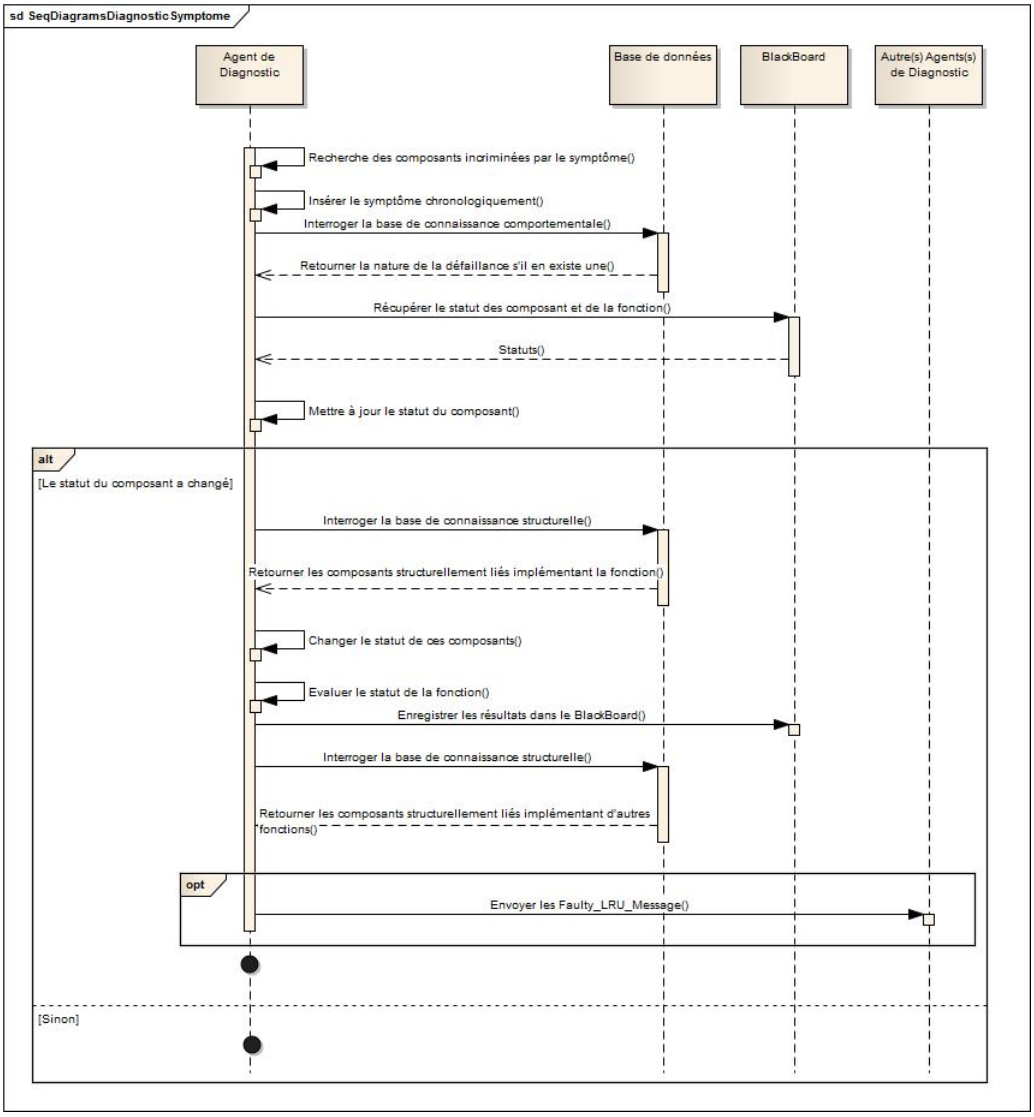


FIGURE 4.4: Diagramme de séquence de la fonction diagnostic dans le cas du traitement d'un symptôme

L'émission d'un « Faulty_LRU_Message » à destination des autres agents de diagnostic nécessite donc qu'un agent de diagnostic puisse traiter ces messages. Lors de la réception d'un « Faulty_LRU_Message », l'agent de diagnostic agit alors selon les étapes suivantes, décrites dans l'algorithme 4.2. Ce processus de traitement d'un « Faulty_LRU_Message » réalisé par un agent de diagnostic peut également être décrit dans l'architecture à l'aide du diagramme de séquence de la figure 4.5 représentant les activités des éléments de l'architecture ainsi que les échanges de messages.

Algorithm 4.2 Algorithme de traitement d'un Faulty_LRU_Message par la fonction diagnostic

Entrées : Faulty_LRU_Message, connaissances MC, MS, MF, MT
Entrée - Sortie : Liste des évaluations des statuts des composants et des fonctions
Début
Rechercher quel est le composant impacté par le «Faulty_LRU_Message»,
Insérer ce « Faulty_LRU_Message » dans la liste chronologique des données reçus incriminant le composant,
Changer le statut du composant ainsi impacté par le «Faulty_LRU_Message» selon la règle suivante :
- passage du statut d'un composant ainsi dépendant à « HS » si son statut était « OK »
- passage du statut d'un composant ainsi dépendant à «HS» si le «Faulty_LRU_Message» a été provoqué par un symptôme daté antérieurement à tous les symptômes du composant ainsi dépendant, que son ancien statut fut «DI» ou «DC»,

Demander à la base de connaissance structurelle la liste des composants implémentant la fonction qui sont structurellement ou topologiquement dépendant du composant.
Changer le statut des composants ainsi dépendant selon la règle suivante :
- passage du statut d'un composant ainsi dépendant à «HS» si son statut était «OK»
- passage du statut d'un composant ainsi dépendant à «HS» si le «Faulty_LRU_Message » a été provoqué par un symptôme dont l'apparition est datée antérieurement à celle de tous les symptômes du composant ainsi dépendant, que son ancien statut fut «DI» ou «DC»,

Evaluer le statut de la fonction,
Inscrire les statuts mis à jour dans le «blackboard»,
Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant d'autres fonctions dont des composants sont dépendants des composants dont le statut a changé.
Envoyer un «Faulty_LRU_Message» par composant des fonctions concernées aux agents en charge de leur diagnostic,
Fin

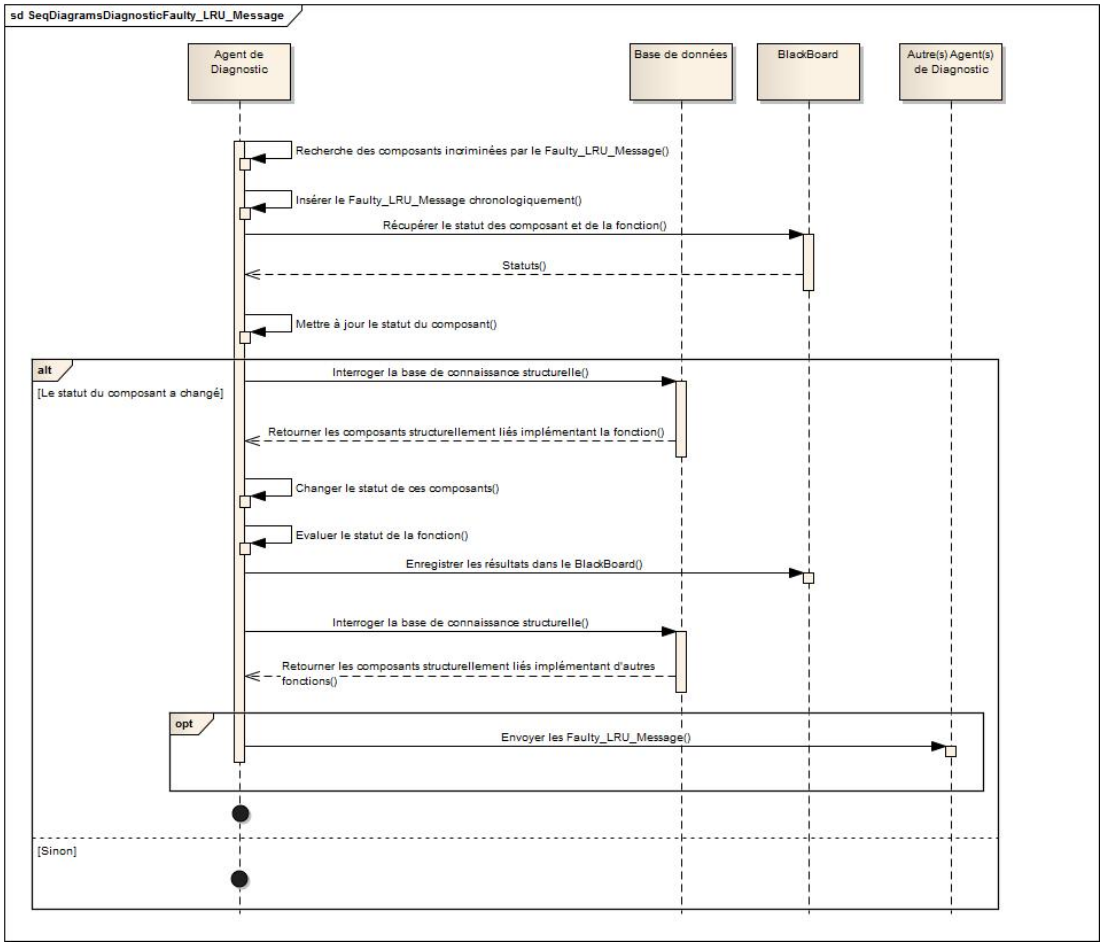


FIGURE 4.5: Diagramme de séquence de la fonction diagnostic dans le cas du traitement d'un Faulty_LRU_Message

Afin que les agents de diagnostic puissent effectuer ces traitements il est nécessaires que les symptômes comme les « Faulty_LRU_Message » comportent certaines informations. La structure de ces messages est représentée sur le schéma de la figure 4.6 pour un symptôme et de la figure 4.7 pour un « Faulty_LRU_Message ». Dans ces données composites que sont les symptômes et les « Faulty_LRU_Message », le champ « Numero_Symptome » est associé par nomenclature à l'identifiant du composant. Le champ « Date_Apparition » d'un symptôme correspond à la date d'apparition du symptôme. Le champ « Date_Detection » correspond à la date à laquelle le symptôme a été détecté et envoyé à l'agent de diagnostic par la couche de détection. Le champ « Raison » d'un « Faulty_LRU_Message » correspond à l'identifiant du composant pour lequel la réception d'un symptôme a entrainé l'émission du message. Les champs « Date_Apparition » et « Date_Detection » d'un « Faulty_LRU_Message » correspondent eux aussi, respectivement, aux

champs « Date_Apparition » et « Date_Detection » du composant pour lequel la réception d'un symptôme a entrainé l'émission du message. Dans ces structures de données, les méthodes sont des accesseurs aux attributs ou des constructeurs de la classe.

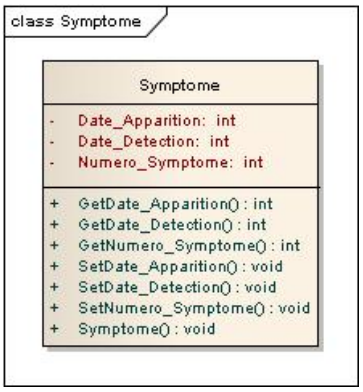


FIGURE 4.6: Structure d'un message du type symptôme

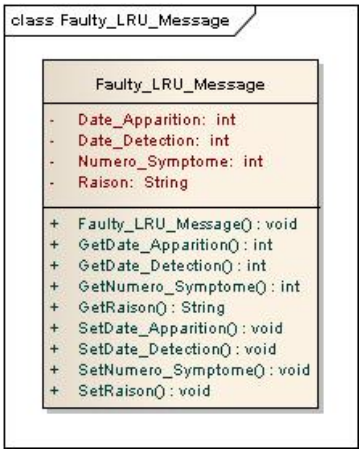


FIGURE 4.7: Structure d'un message du type Faulty_LRU_Message

4.2.3. Déploiement distribué du pronostic

Dans le principe de distribution que nous avons exposé, le principe de pronostic exposé dans le chapitre 3 est distribué aux agents de pronostic. Tous les agents de pronostic effectuent le même traitement. Cependant, ils sont dédiés à des fonctions du STCGD différentes. Lorsqu'un agent de pronostic reçoit un RUL de la couche

de surveillance, il agit conformément à l'algorithme 4.3. Ce processus, mené par un agent de pronostic, peut également être décrit dans l'architecture à l'aide du diagramme de séquence de la figure 4.8 représentant les activités des éléments de l'architecture ainsi que les échanges de messages.

Algorithm 4.3 Algorithme de traitement d'un RUL par la fonction pronostic

Entrées : RUL, connaissances MS, MF, MT
Entrées - Sorties : Liste des évaluations des TBAB des composants et des fonctions
Début
  Mettre à jour la liste des RUL des composants de la fonction que l'agent est chargé de pronostiquer. Cette mise à jour concerne le TBAB du composant concerné par le RUL envoyé par la couche de surveillance,
  Si la durée du RUL envoyé par la couche de surveillance est inférieure à la valeur du TBAB correspondant au même composant,
    Affecter à la durée du TBAB du composant concerné la valeur de la durée du RUL envoyé par la couche de surveillance,
    Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant la fonction qui sont structurellement ou topologiquement dépendant du composant dont le TBAB vient d'être modifié
    Remplacer la durée des TBAB de la liste des composants fournit par la base de connaissances, si elle est supérieure à la durée du RUL reçu, par la durée de ce RUL,
    Evaluer le TBAB de la fonction dont il a le pronostic en charge,
    Inscrire les nouvelles valeurs pour les RUL et TBAB dans le « blackboard »
    Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant d'autres fonctions dont des composants sont dépendants des composants dont le TBAB a changé.
    Envoyer un « TBAB_message » par composant des fonctions concernées aux agents en charge de leur pronostic,
  Fin Si
Fin

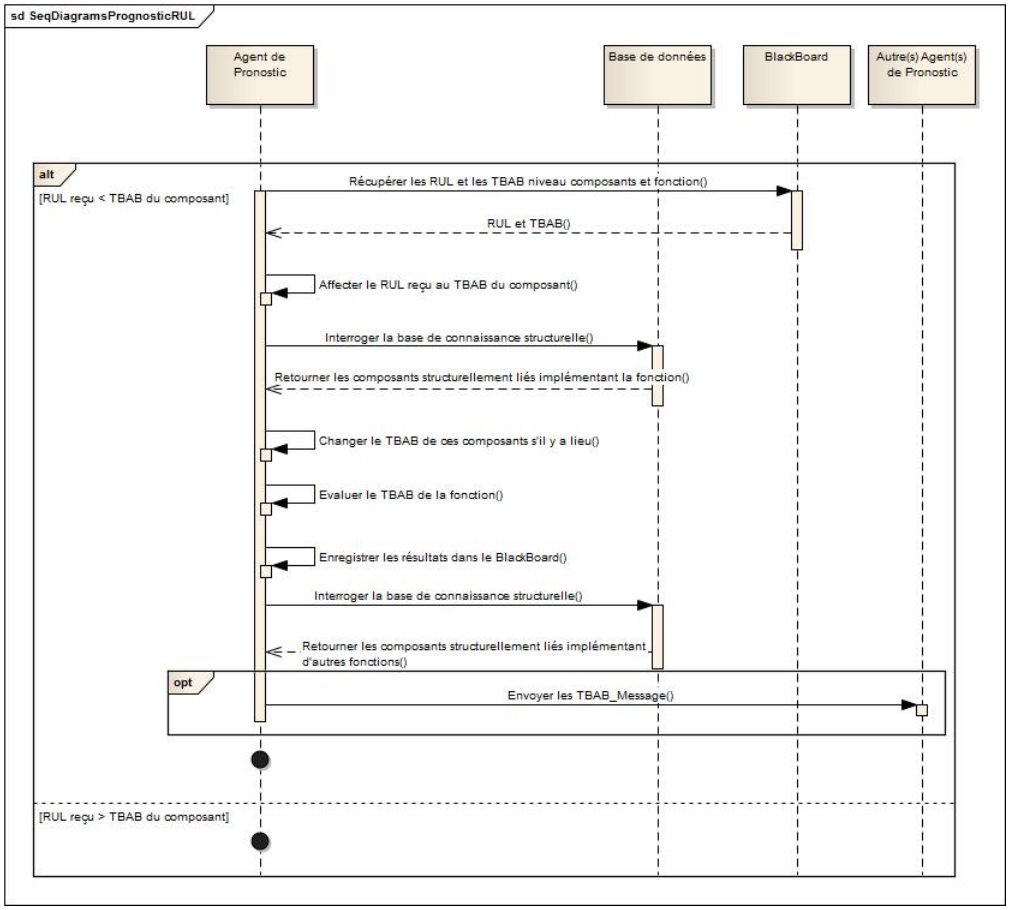


FIGURE 4.8: Diagramme de séquence de la fonction pronostic dans le cas du traitement d'un RUL

L'émission d'un « TBAB_message » à destination des autres agents de pronostic nécessite donc qu'un agent de pronostic puisse traiter ces messages. Lors de la réception d'un « TBAB_message » l'agent de pronostic agit alors selon les étapes suivantes décrites dans l'algorithme 4.4. Ce processus de traitement d'un « TBAB_message », réalisé par un agent de pronostic, peut également être décrit dans l'architecture à l'aide du diagramme de séquence de la figure 4.9 représentant les activités des éléments de l'architecture ainsi que les échanges de messages.

Algorithm 4.4 Algorithme de traitement d'un TBAB_Message par la fonction pronostic

Entrées : TBAB_Message, connaissances MS, MF, MT
Entrées - Sorties : Liste des évaluations des TBAB des composants et des fonctions
Début
  Si la valeur du champ TBAB du « TBAB_Message » envoyé par la couche de surveillance est inférieure à la valeur du TBAB correspondant au même composant,
    Affecter à la durée du TBAB du composant concerné la valeur du champ TBAB du « TBAB_Message » reçu,
    Demander à la base de connaissance (structurelle et topologique) la liste des composants implémentant la fonction qui sont structurellement ou topologiquement dépendant du composant dont le TBAB vient d'être modifié
    Remplacer la durée des TBAB des composants de la liste fournie par la base de connaissances, si elle est supérieure à la valeur du champ TBAB du «TBAB_Message» reçue, par cette valeur,
    Evaluer le TBAB de la fonction dont il a le pronostic en charge,
    Inscrire les nouvelles valeurs des TBAB des composants et de la fonction dans le « blackboard »,
    Demander à la base de connaissances (structurelles et topologiques) la liste des composants implémentant d'autres fonctions dont des composants sont dépendants des composants dont le TBAB a changé.
    Envoyer un « TBAB_message » par composant des fonctions concernées aux agents en charge de leur pronostic,
  Fin Si
Fin

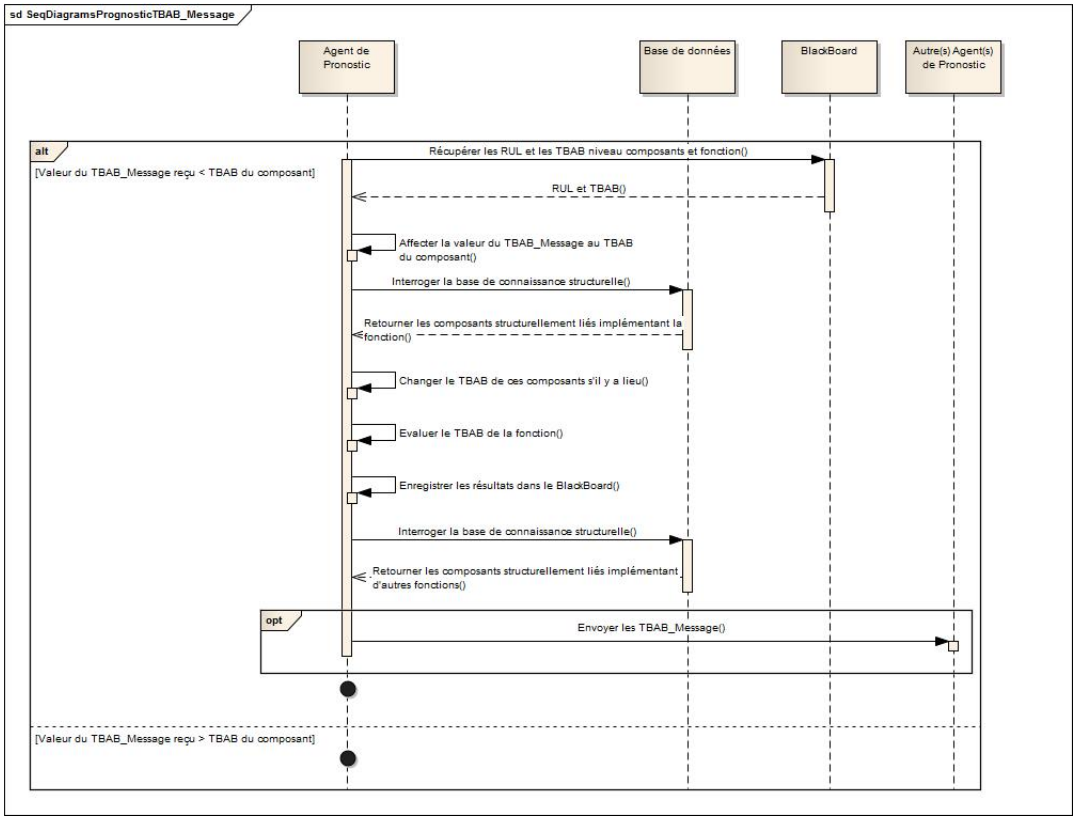


FIGURE 4.9: Diagramme de séquence de la fonction pronostic dans le cas du traitement d'un TBAB_Message

Lors de ces processus de pronostic, tout TBAB de composant ou de fonction est associé à l'identifiant du composant pour lequel l'émission du RUL a provoqué le changement vers ce TBAB. Afin que les agents de pronostic puissent effectuer ces traitements il est nécessaire que les RUL comme les « TBAB_message » comportent certaines informations. La structure de ces messages est représentée sur le schéma des figures 4.10 pour un RUL et 4.11 pour un « TBAB_Message ».

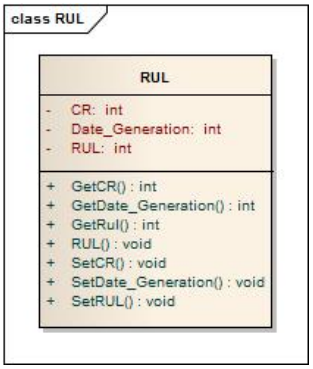


FIGURE 4.10: Structure d'un message du type RUL

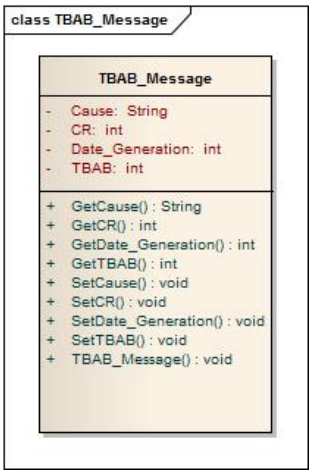


FIGURE 4.11: Structure d'un message du type TBAB_Message

Dans ces données composites que sont les RUL et les « TBAB_Message », le champ « CR » est associé par nomenclature à l'identifiant du composant sur lequel s'applique le RUL. Ce RUL est défini dans le champ « RUL » d'un objet de type RUL et dans le champ « TBAB » d'un objet de type TBAB. Le champ « Date_Generation » d'un RUL correspond à la date à laquelle la couche de détection a réévalué et envoyé le RUL du composant concerné. Le champ « Cause » d'un « TBAB_Message » correspond à l'identifiant du composant pour lequel la réception d'un RUL ou d'un « TBAB_Message » a entrainé l'émission du message. Le champ « Date_Generation » d'un « TBAB_Message » correspond au champ « Date_Generation » du composant pour lequel la réception d'un RUL ou d'un « TBAB_Message » a entrainé l'émission du message. Dans ces structures de données, les méthodes sont des accesseurs aux attributs ou des constructeurs de la classe.

4.2.4. Construction du HA

Les agents de diagnostic et de pronostic mettent à jour les informations contenues dans le « blackboard ». Ces informations concernent tant les statuts des composants et fonctions que leur RUL et/ou TBAB. Ces informations constituent donc le HA du STCGD. Cette construction distribuée du HA doit, notamment à des fins d'évaluation de performances, être comparée à une construction centralisée.

4.3. Déploiement centralisé du diagnostic et du pronostic d'un STCGD

Le déploiement centralisé envisagé afin d'évaluer l'intérêt de la distribution proposée de la construction du HA par des agents de diagnostic et des agents de pronostic opérant sur des fonctions différentes du STCGD consiste en une implémentation d'un unique agent de diagnostic et d'un unique agent de pronostic. Dans cette structure, représentée sur le schéma de la figure 4.12, il y a toujours :

- un élément représentant la couche de surveillance,
- un élément représentant la base de connaissance (BdD),
- un élément représentant le « blackboard »,
- un agent de diagnostic AD fonctionnant selon le principe de diagnostic décrit dans le chapitre 3,
- un agent de pronostic AP fonctionnant selon le principe de pronostic décrit dans le chapitre 3.

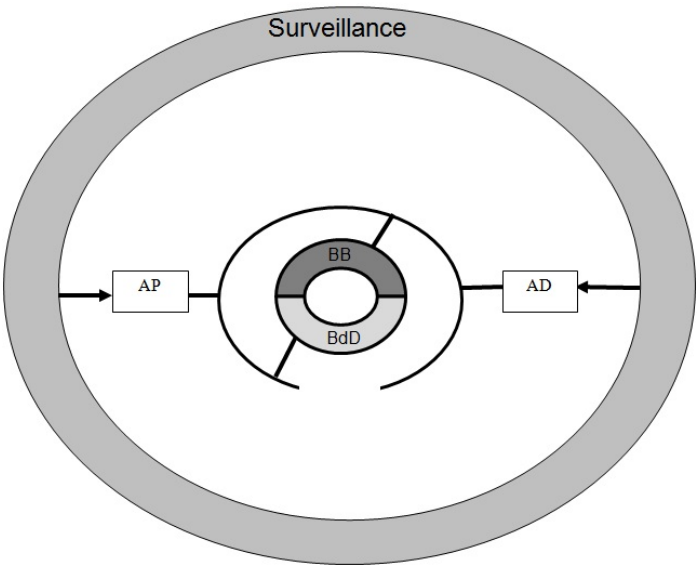


FIGURE 4.12: Déploiement centralisé d'une fonction d'évaluation de l'état de santé d'un STCGD

Pour le diagnostic et le pronostic, les activations des éléments de l'architecture centralisée et les échanges de messages sont identiques à ceux de l'architecture distribuée. Cependant, étant donné qu'un seul et unique agent assure la fonction de diagnostic (ou de pronostic), lors de l'interrogation de la base de données, tous les composants ayant des dépendances structurelles (ou topologiques) seront retournés, toutes fonctions confondues, et pas seulement ceux contribuant à l'implémentation de la fonction dont un des composants vient d'être modifié (statut ou TBAB). Les modes de communication sont les mêmes que ceux décrits dans le cas du déploiement distribué. Bien évidemment dans ce déploiement aucune communication n'est réalisée entre agents de diagnostic ou de pronostic puisqu'ils sont uniques. L'activité de diagnostic, enclenchée lors de la réception d'un symptôme, peut alors être décrite par le diagramme de séquence de la figure 4.13. L'activité de pronostic, enclenchée lors de la réception d'un RUL, peut alors être décrite par le diagramme de séquence de la figure 4.14.

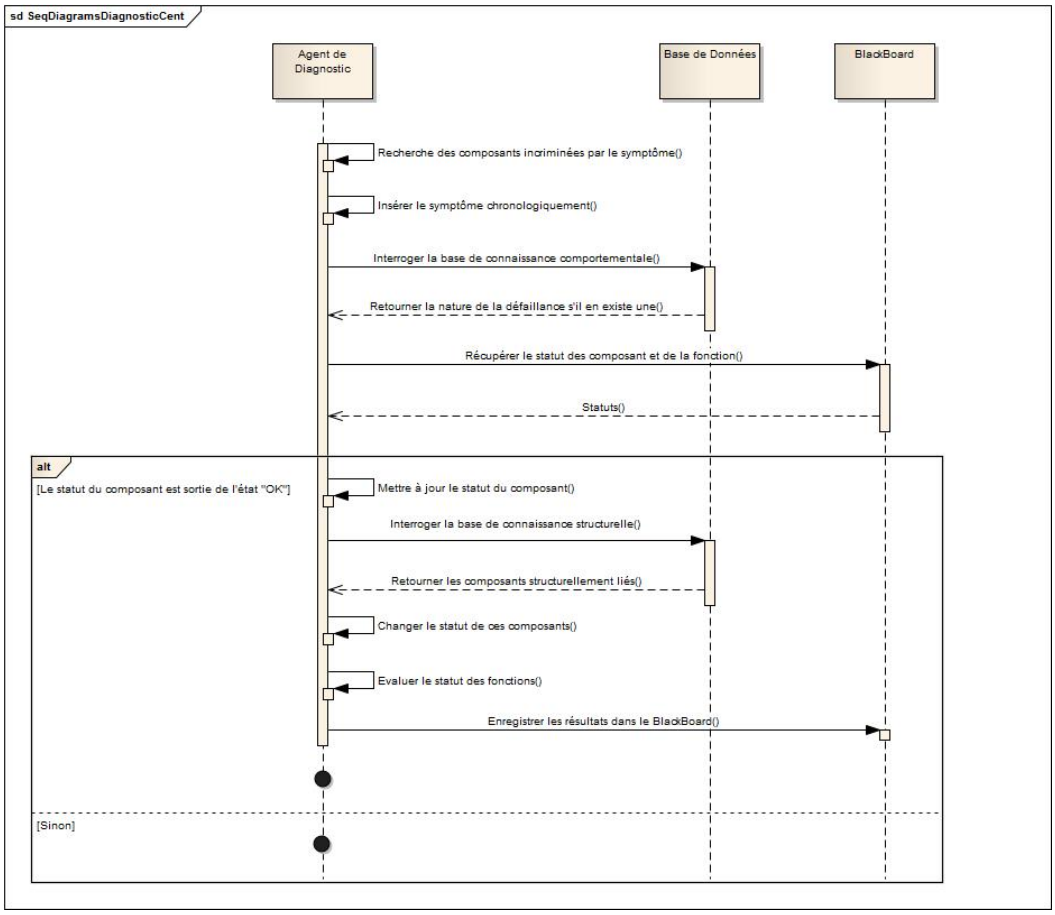


FIGURE 4.13: Diagramme de séquence de la fonction diagnostic déployée de façon centralisé

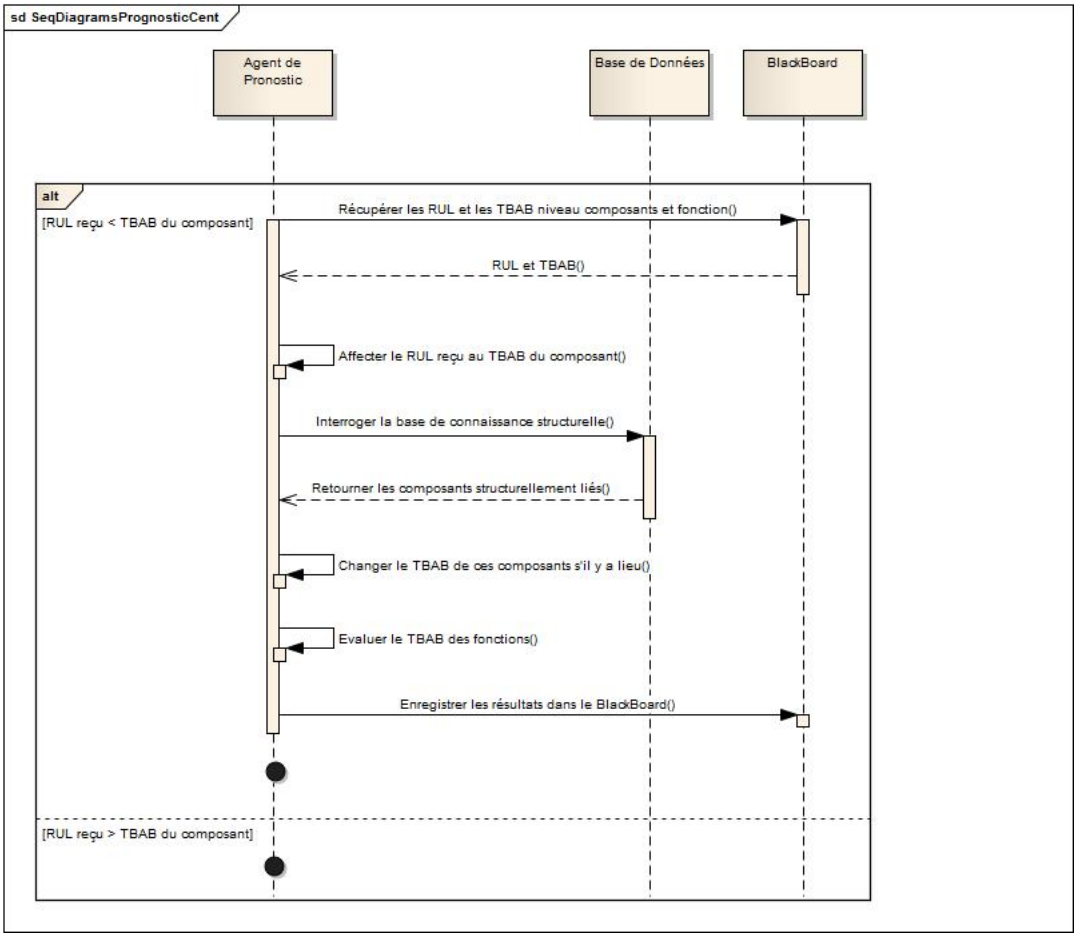


FIGURE 4.14: Diagramme de séquence de la fonction pronostic déployée de façon centralisé

Afin d'obtenir les résultats les plus probants possibles pour les comparaisons de performances entre les implémentations distribuées et centralisées du diagnostic et du pronostic, nous avons opté pour une plateforme de simulation. En effet, une plateforme de simulation configurable permet de représenter différents STCGD et de générer rapidement des scénarios pour le pronostic et le diagnostic. La plateforme de simulation permet également d'obtenir ces comparaisons à des coûts très nettement inférieurs à des tests pratiqués sur des STCGD réels et sans risques sur les biens, les personnes et l'environnement.

4.4. Plateforme d'évaluation

4.4.1. Objectifs de la plateforme

Le développement d'une plateforme de simulation a pour objectif l'évaluation et la comparaison de différents déploiements possibles des fonctions de diagnostic et de pronostic pour les STCGD. Cette plateforme doit permettre de représenter différents types de STCGD. Cela est assuré en renseignant la connaissance qui a été décrite à la section 2.5. La plateforme doit permettre de générer des scénarios pour le pronostic et le diagnostic mais aussi assurer les différents modes de communication pouvant entre mis en œuvre (synchrone et asynchrone). Finalement, et pour répondre à l'objectif de comparaison, la plateforme doit évidemment permettre de tester différents déploiements dont ceux dont nous avons fait la description dans les sections 4.2 et 4.3. La comparaison des différents déploiements possibles ne peut être assurée que par le biais de critères évalués pour chacun d'eux. Les critères que nous avons retenus sont les suivants :

- Le nombre d'objets traités : le nombre d'objets traités sert de base pour tracer l'évolution des autres indicateurs cités plus loin, en faisant évoluer le nombre d'objets traités (symptôme, et «Faulty_LRU_Message» pour le diagnostic, RUL et «TBAB_Message» pour le pronostic).
- Le temps de convergence : ce temps, représentant la durée totale entre l'envoi du premier élément (symptôme ou RUL) par l'objet simulant la couche de surveillance et la fin du processus de diagnostic ou de pronostic (arrêt du dernier agent de diagnostic ou de pronostic, synonyme de fin de session), est évalué afin de déterminer l'incidence du nombre d'objets traités sur ce temps.
- La charge de calcul : elle est calculée pour chacun des éléments de l'architecture et plus particulièrement pour les agents de diagnostic et de pronostic. Cette charge représente le rapport entre le temps d'occupation des agents et le temps total de simulation. Ce rapport dépend de la densité d'émission des données.

4.4.2. Développement de la plateforme

Le développement de la plateforme est basé sur le concept orienté objet. Un objet peut être qualifié de la façon suivante : un objet est quelque chose de réel ou d'abstrait, de tangible ou d'intangible, à propos duquel on emmagasine des données et des méthodes (procédures, fonctions) permettant de les manipuler (Bersini, 2008; Meyer et Jouvelot, 2008). Cette définition contient trois des éléments fonda-

mentaux qui définissent un objet soit, le caractère réel ou abstrait d'un objet, le caractère intangible ou tangible et ce qu'il contient (les données et les méthodes de manipulation). Un quatrième élément fondamental de l'objet est son unicité. Chaque objet est une occurrence et peut donc être identifié de façon unique, être distingué des autres objets. La structure de données d'un objet est définie en termes d'attributs. Ces attributs sont des valeurs qui sont associées et étroitement connectées aux objets (Meyer et Jouvelot, 2008). Un autre concept de l'orienté objet porte sur la notion d'héritage. Comme dans le monde réel, certains objets sont associés entre eux. Grâce au mécanisme d'héritage, une sous-classe peut hériter des méthodes et des propriétés de sa classe parente. Les classes héritées ne sont au départ que des copies de leur parent, mais peuvent ensuite évoluer en ajoutant des propriétés ou des méthodes, voire en modifiant les méthodes existantes (mécanisme de surcharge). L'héritage est un concept fondamental de la programmation par objets (Debrauwer et Van Der Heyde, 2008). L'héritage consiste à définir une classe d'objets à partir d'une autre classe tout en l'affinant de deux façons :

- en ajoutant des attributs,
- en ajoutant des méthodes ou en redéfinissant des méthodes déjà définies dans la surclasse. (surcharge)

Ainsi, l'approche objet permet de réutiliser des méthodes déjà développées par l'utilisation du mécanisme d'héritage et permet de définir plusieurs objets différents implémentant les mêmes méthodes. Ce dernier point sera très utile au déploiement distribué de la fonction diagnostic. Différents langages permettent une programmation dans l'approche orientée objet comme le C++, Java, Eiffel, SmallTalk ou bien encore le C#. Nous avons choisi le C# car il a été créé afin de permettre une utilisation optimale des fonctionnalités de la bibliothèque .NET. Cette bibliothèque, considérée comme un middleware, contient des méthodes facilitant le développement d'applications nécessitant des communications par messages comme dans l'approche client/serveur. Cette bibliothèque permet de gérer des fichiers XML en création, lecture et écriture mais aussi des communications synchrones et asynchrones. Cet outil répond donc en tout point aux contraintes de la plateforme. Le développement a été effectué à l'aide de Visual Studio. La communication entre agents est basée sur le protocole TCP/IP mettant en œuvre la carte réseau de l'ordinateur. Le choix de ce protocole est motivé par la possibilité offerte de déployer les différents agents sur des ordinateurs différents.

Nous considérons la proposition du principe de pronostic faite au chapitre 3 et son déploiement distribué ou centralisé comme un résultat. Le pronostic de système est très rarement déployé et le pronostic dynamique tel que nous le pro-

posons l'est encore moins. Contrairement au principe de diagnostic décrit à la section 3.3, le principe de pronostic est monotone. En effet, les TBAB ne sont réévalués que si des RUL leurs sont inférieurs. De plus, la fréquence d'émission de RUL est généralement basse. Le processus de détermination des TBAB n'est pas critique puisqu'il est utilisé principalement pour l'aide à la maintenance et à la planification. Il n'y a donc pas nécessairement d'intérêt majeur, dans le cadre du pronostic, à comparer les performances d'un déploiement distribué à celles d'un déploiement centralisé. Toutefois, il n'en est pas de même dans le cas du diagnostic. Les méthodes mises en œuvre sont nombreuses et les déploiements différents comme nous l'avons mentionné dans le chapitre 1. De plus les résultats produits par une fonction de diagnostic sont à la fois destinés à l'aide à la maintenance et à la planification et destinés à la conduite du système notamment lorsque des fonctions sont défaillantes amenant à des modes de fonctionnement dégradés, des révisions d'objectifs ou des arrêts. De ce point de vue, le diagnostic est plus critique que le pronostic. Le principe de diagnostic que nous proposons est non monotone et il est donc nécessaire de vérifier qu'il fournit les résultats attendus. C'est pour ces raisons que nous nous attachons, particulièrement, à fournir des résultats sur les comparaisons de performances, à l'aide de la plateforme, entre le déploiement distribué et centralisé du principe de diagnostic présenté à la section 3.3.

4.4.3. Mise en œuvre de la plateforme

Dans le cas du diagnostic, la plateforme permet la génération de symptômes afin de simuler une couche de détection. La génération des symptômes est aléatoire. A chaque itération, le générateur décide, en fonction d'un premier seuil, du tirage d'un symptôme. Si le tirage d'un symptôme est décidé, le générateur recherche si le composant associé au précédent symptôme qu'il a tiré présente des dépendances structurelles ou topologiques avec d'autres composants. Pour cela, le générateur dispose d'une représentation des connaissances structurelles et topologiques. Si c'est le cas le générateur décide en fonction d'un second seuil de tirer ou non un symptôme associé à un composant ainsi dépendant. Si c'est le cas un tel symptôme est tiré aléatoirement parmi la liste des symptômes associés à des composants ainsi dépendants. Dans les autres cas pour lequel un tirage de symptôme a été décidé, un symptôme est tiré aléatoirement. Un symptôme tiré ne peut plus l'être. Ceci correspond alors à des tirages sans remise. Le champ « Date_Detection » du symptôme prend alors pour valeur la date à laquelle son tirage a été effectué. Ensuite, le générateur transmet le symptôme, dans le cas du déploiement centralisé, à l'agent de diagnostic ou, dans le cas du déploiement distribué, à l'agent de

diagnostic en charge de la fonction à laquelle le composant associé au symptôme concourt. Le nombre de symptômes générés varie en fonction des connaissances et donc du STCGD représenté. La densité des émissions des symptômes varie en fonction du premier seuil. Un second seuil fixé à une valeur élevée (proche de 1) permet de renforcer l'aspect défaillance en cascade.

Le champ « Date_Apparition » du symptôme est une date antérieure à celle du champ « Date_Detection ». Cette date peut être issue de la datation des mesures ayant servi à élaborer le symptôme. Si cette datation des mesures n'est pas effectuée, cette date est identique à celle du champ « Date_Detection ». Dans le cas de la comparaison de déploiements, la différence de valeur entre les champs « Date_Apparition » et « Date_Detection » n'est pas pertinente puisque dans le cas du diagnostic seul, le champ « Date_Apparition » est utilisé pour évaluer les statuts des composants et fonctions. Le générateur affecte donc systématiquement la valeur du champ « Date_Detection » au champ « Date_Apparition ». Les tirages aléatoires permettent de représenter le fait que, dans les STCGD, les symptômes ne sont pas forcément révélés dans l'ordre chronologique de l'apparition des défaillances.

Les symptômes envoyés par le générateur sont tracés dans un fichier XML dont la structure des enregistrements est représentée par le schéma XML de la figure 4.15. Les enregistrements des fichiers traçant la réception des symptômes par les agents sont constitués selon le même schéma.

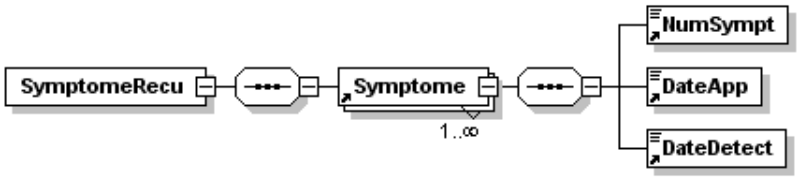


FIGURE 4.15: Structure des enregistrements des fichiers traçant les symptômes (reçus ou envoyés)

Afin d'éviter des erreurs liés lors de la duplication des connaissances structurelles et comportementales, nous avons choisi que le générateur de symptômes exploite la base de données contenant ces connaissances également utilisée par l'agent ou les agents de diagnostic. Cette base de données permet les accès aux fichiers XML représentant les connaissances dont la structure des enregistrements a été décrite dans le chapitre 2. Une interface permet la saisie des connaissances dans les différents fichiers.

Enfin, l'agent de diagnostic, dans le cas du déploiement centralisé, ou les agents de diagnostic, dans le cas du déploiement distribué, sont implémentés ainsi que le « blackboard ». Le fichier XML du « blackboard » contenant les statuts des composants et fonctions est vierge lors de l'initialisation des simulations. C'est-à-dire que tous les composants et fonctions sont dans le statut « OK ». Dans la plateforme, l'évolution du fichier XML du « blackboard » est tracée dans le but de validation du principe de diagnostic que le déploiement soit centralisé ou distribué.

4.5. Evaluation et comparaison des déploiements

Afin d'évaluer et comparer les déploiements, nous utilisons un cas d'étude.

4.5.1. Cas d'étude

Le cas d'étude servant à l'évaluation et à la comparaison des déploiements représente un STCGD composé de trois systèmes implémentant chacun une fonction. Chacune de ces fonctions est implémentée par trois composants. Une de ces fonctions est une fonction de redondance qui est assurée si au moins deux des trois composants qui l'implémentent fonctionnent. Les trois fonctions sont structurellement liées entre elles par des liaisons entre les composants les implémentant. Ce cas d'étude est représenté sur le schéma de la figure 4.16 où les dépendances structurelles sont représentées par les flèches en pointillés et où les connaissances fonctionnelles y sont représentées par les encadrés. Les connaissances topologiques ne sont pas considérées dans ce cas d'étude. En effet, elles n'apportent que peu sur le plan de la complexité du STCGD représenté si ce n'est qu'un parcours de graphe supplémentaire lors d'une requête sur cette connaissance. Un tel parcours de graphe est déjà effectué pour une requête concernant les connaissances structurelles.

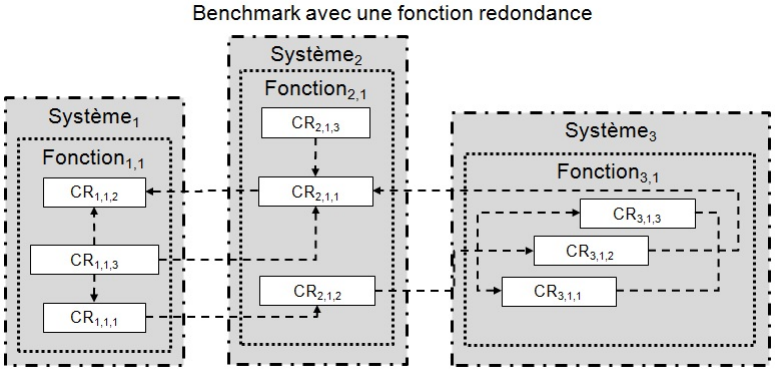


FIGURE 4.16: Cas d'étude

Chaque composant peut révéler trois symptômes différents. Vingt sept symptômes peuvent donc être générés en tout. La connaissance comportementale, comme elle a été décrite à la section 2.5.4, est constituée d'association de symptômes émanant d'un même composant identifié à une cause identifiée. La connaissance comportementale de notre cas d'étude sera ainsi constituée de 7 comportements identifiés comme défaillant comme présentée sur le tableau 4.1.

Cause identifiée	Numéro de composant	Numéro des symptômes
Cause défaillance $CR_{1,1,1}$	$CR_{1,1,1}$	$S_{1,1,1,1}; S_{1,1,1,2}; S_{1,1,1,3}$
Cause défaillance $CR_{3,1,2}$	$CR_{3,1,2}$	$S_{3,1,2,1}; S_{3,1,2,2}$
Cause 1 défaillance $CR_{2,1,1}$	$CR_{2,1,1}$	$S_{2,1,1,1}$
Cause 2 défaillance $CR_{2,1,1}$	$CR_{2,1,1}$	$S_{2,1,1,2}$
Cause 3 défaillance $CR_{2,1,1}$	$CR_{2,1,1}$	$S_{2,1,1,3}$
Cause défaillance $CR_{3,1,1}$	$CR_{3,1,1}$	$S_{3,1,1,2}$
Cause défaillance $CR_{2,1,2}$	$CR_{2,1,2}$	$S_{2,1,2,2}; S_{2,1,2,3}$

TABLE 4.1: Tableau récapitulatif de la connaissance comportementale identifiée pour notre cas d'étude

Il est bien sûr possible d'ajouter d'autres causes connues de défaillances mais cela n'a aucun intérêt dans notre cas d'étude. En effet, le seul effet est l'augmentation du temps de traitement de l'algorithme de diagnostic. L'objectif est de vérifier le bon comportement de l'algorithme de diagnostic à identifier les causes connues de défaillances en utilisant convenablement la connaissance comportementale.

4.5.2. Résultats

Le cas d'étude semble de trop petite dimension pour conclure sur l'intérêt d'un déploiement centralisé ou distribué du principe de diagnostic proposé. Toutefois, il nous a permis de valider ce principe de diagnostic ainsi que sa convergence quelque soit le déploiement. Ce travail a été effectué par l'analyse des résultats de diagnostic et des fichiers de traçage des symptômes reçus et envoyés de chaque agent, et des différentes connaissances saisies concernant le système étudié (MC,MF, MS).

Afin d'identifier le déploiement le plus adapté à un STCGD, le générateur de symptôme a été modifié afin que les tirages de symptômes soient effectués avec remise. L'agent ou les agents de diagnostic sont alors actifs tant que le dernier symptôme transmis par le générateur n'a pas été traité. De plus le générateur n'effectue plus le tirage à chaque itération pour savoir si lors de cette itération il génère

un symptôme. Un symptôme est généré à chaque itération. Chaque symptôme est ensuite ajouté à une liste. Lorsque le nombre de symptômes dans la liste atteint un nombre prédéterminé. Les symptômes sont transmis dans un mode asynchrone les uns à la suite des autres à l'agent ou aux agents de diagnostic. Cela permet de charger au maximum les agents de diagnostic et de se trouver dans des cas très défavorables. Sur les courbes de la figure 4.17, les vitesses de convergence du diagnostic en secondes en fonction du nombre de données traitées sont représentées. La courbe bleue présente le cas d'un déploiement centralisé et la courbe rouge le déploiement distribué. Le tableau 4.2 présente les valeurs qui ont permis de construire la courbe relative au déploiement centralisé. Le tableau 4.3 présente les valeurs qui ont permis de construire la courbe relative au déploiement distribué. Nous précisons que ces résultats sont obtenus pour une implémentation sur une unique plateforme de calcul (un seul ordinateur) des éléments (agent(s), générateur de symptôme, base de données et « blackboard ») du déploiement centralisé ou distribué.

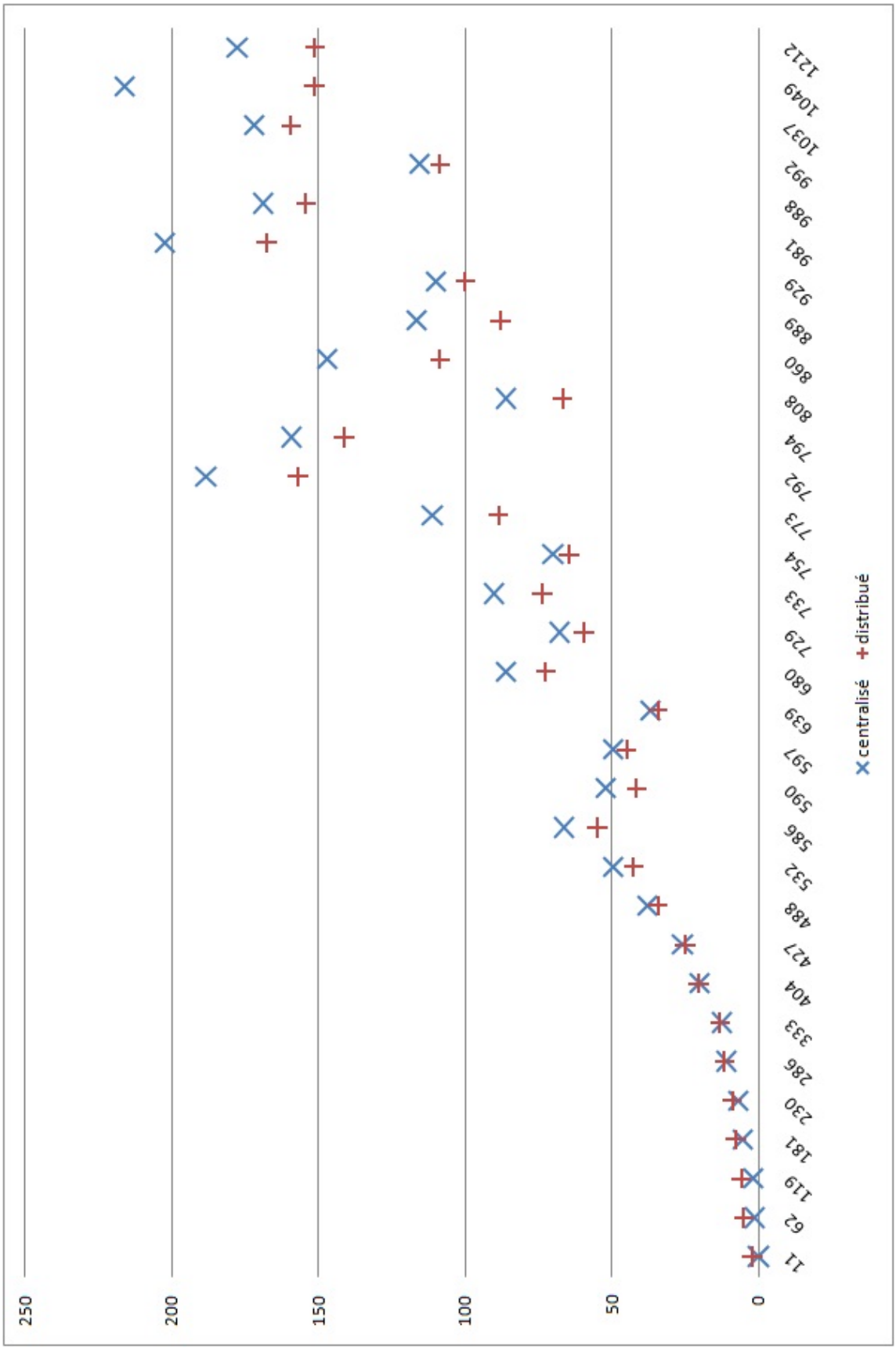


FIGURE 4.17: Vitesse de convergence (en secondes) d'un déploiement centralisé et distribué de la fonction de diagnostic en fonction du nombre de données traitées

Nombre de symptomes envoyés	Nombre de données traitées	Nombre de données retraitées	Tps de convergence (s)
10	11	7	0,1800002
50	62	149	1,3120031
100	119	463	2,0440438
150	181	1241	5,4781492
200	230	2246	7,1272347
250	286	4220	10,9762858
300	333	3923	12,5423821
350	404	6723	20,0065233
400	427	8011	25,993618
450	488	17471	37,8028839
500	532	13291	49,7622572
550	586	19708	66,4297892
550	590	17952	52,2258289
550	597	16594	49,679229
600	639	12823	37,0371966
650	680	22844	86,154399
700	729	27240	67,7754555
700	733	36065	90,382387
700	754	28347	70,1251646
750	773	35043	111,3246485
750	792	57500	188,4255982
750	794	50644	159,3625703
700	808	27438	86,2127049
800	860	55051	147,1474262
850	889	39560	116,5108643
900	929	48838	109,9123551
950	981	68268	202,3071356
950	988	52189	168,8846734
950	992	53750	115,6371608
1000	1037	62941	171,9355361
1000	1049	81039	216,1245476
1050	1212	62432	177,7928593

TABLE 4.2: Tableau relatif au résultat de l'évaluation de la vitesse de convergence d'un déploiement centralisé

Nombre de symptomes envoyés	Nombre de données traitées	Nombre de données retraitées	Tps de convergence (s)
10	11	6	2,1420053
50	62	143	5,0880288
100	119	466	5,8920786
150	181	1218	7,6511754
200	230	2380	8,8692842
250	286	4284	11,5943914
300	333	4033	13,2034517
350	404	7622	20,3408158
400	427	8339	25,0179338
450	488	17818	34,26535
500	532	13886	42,66573
550	586	20405	54,8634311
550	590	18960	41,6416999
550	597	16405	44,9945318
600	639	13509	34,2275643
650	680	24293	72,5001563
700	729	27302	59,4695785
700	733	37731	73,7620611
700	754	28408	64,5542644
750	773	34946	88,6699691
750	792	59953	156,9380353
750	794	51534	141,2762058
700	808	30676	66,838213
800	860	55081	108,6260314
850	889	40608	87,9330006
900	929	50037	99,97725
950	981	69461	167,6521007
950	988	52339	154,3309853
950	992	54491	108,6683436
1000	1037	63364	159,3289525
1000	1049	83615	151,327419
1050	1212	68538	151,2029448

TABLE 4.3: Tableau relatif au résultat de l'évaluation de la vitesse de convergence d'un déploiement distribué

D'après les résultats présentés sur les courbes de la figure 4.17, le déploiement centralisé du principe de diagnostic proposé converge plus rapidement que son

déploiement distribué si le nombre de données traitées est inférieur à une valeur de l'ordre de 350. Les vitesses de convergence des deux déploiements sont presque identiques dans le cas où le nombre de données traitées est compris entre des valeurs de l'ordre de 350 à 450. Au-delà d'un nombre de données traitées supérieur à une valeur de l'ordre de 450, la vitesse de convergence du déploiement distribué est la plus rapide. Nous constatons également que l'écart entre les vitesses de convergence des deux déploiements semble tendre, en moyenne, à peu s'accroître voire à stagner. Cela peut être dû au fait que tous les éléments de l'architecture sont implémentés sur la même plateforme de calcul. Nous supposons que cela peut correspondre, pour un grand nombre de données traitées, à effectuer un déploiement centralisé. En effet, cette implémentation mono plateforme ne permet pas de bénéficier du parallélisme des traitements réalisés par les agents du déploiement distribué implémentés sur des plateformes différentes. L'écart entre les 2 courbes dépend de l'équilibre des charges des agents de diagnostic dans le cas d'un déploiement distribué. En effet, si dans un déploiement distribué, seul un agent assure le diagnostic de toutes les données générées, alors, cela revient à un comportement similaire à un déploiement centralisé. Ainsi, si tous les agents de diagnostic d'une architecture distribuée ont une charge proche de 100%, cela signifie que l'apport du déploiement distribué est optimal. La figure 4.18 présente l'évolution de la moyenne et de l'étendue des charges des agents de diagnostic dans le cas d'un déploiement distribué de la fonction en fonction du nombre de données traitées. Le tableau 4.4 présente les valeurs qui ont permis de construire les courbes relatives à la moyenne et à l'étendue des charges des agents de diagnostic d'un déploiement distribué de la fonction de diagnostic.

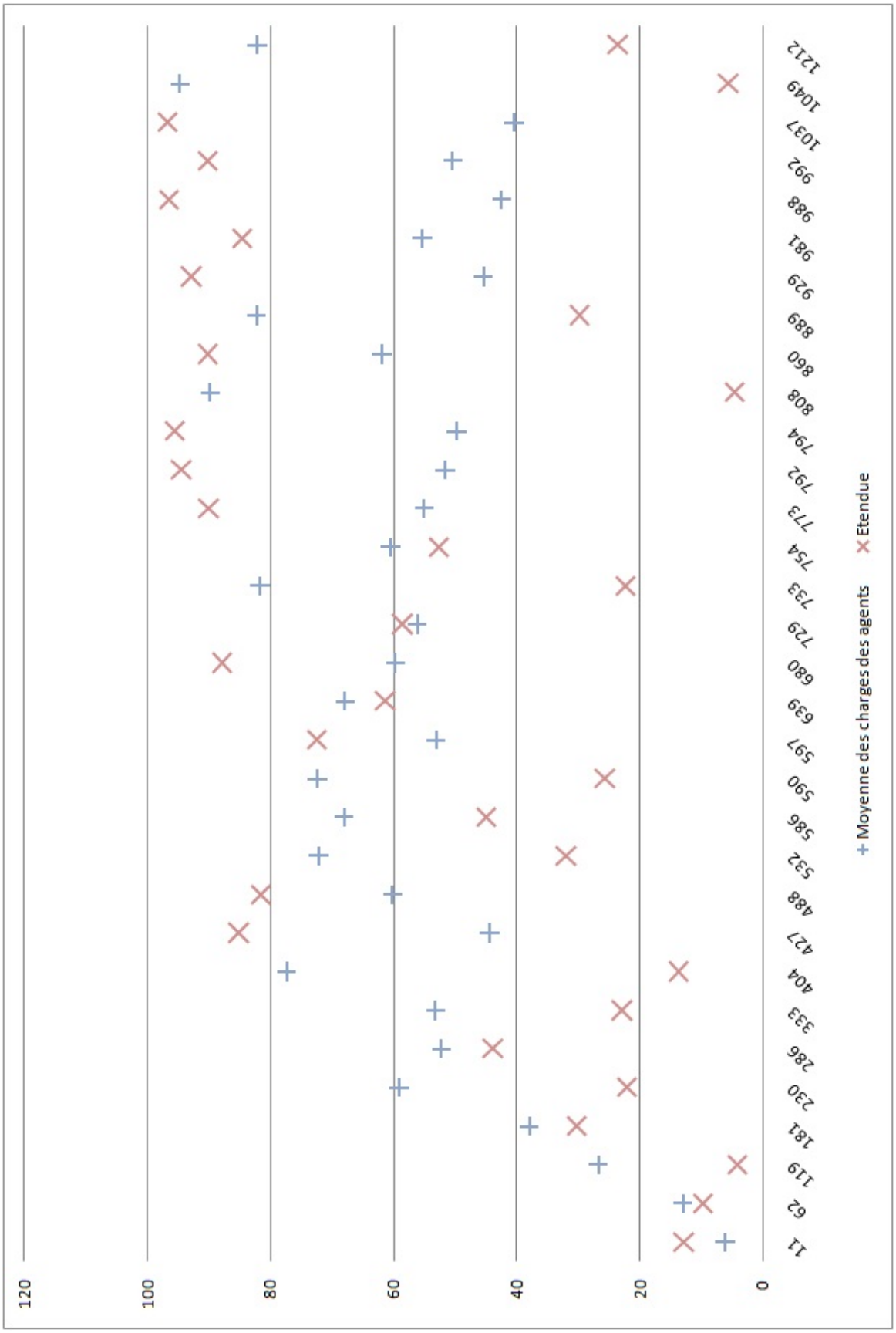


FIGURE 4.18: Evolution de la moyenne et de l'étendue des charges des agents de diagnostic d'une architecture déployée en fonction du nombre de données traitées

Nombre de données traitées	Moyenne des charges des agents	Etendue
11	6,131354266	12,8851829
62	12,96515067	9,66980179
119	26,69751385	4,12382313
181	37,94302333	30,244519
230	59,03410747	22,0540142
286	52,20552614	43,9455761
333	53,1426281	22,9342036
404	77,30953888	13,6284721
427	44,40073384	85,1862335
488	60,16535528	81,6486597
532	72,12918002	31,9729167
586	68,04323946	44,9643373
590	72,36597827	25,7343961
597	53,09601836	72,4310668
639	67,89411441	61,3861641
680	59,70981739	87,8202625
729	56,10693598	58,6374697
733	81,73469875	22,2801528
754	60,44116393	52,614468
773	55,05917753	90,0898557
792	51,63270273	94,5179974
794	49,75417922	95,5383896
808	89,8089745	4,64869071
860	61,88725811	90,1443077
889	82,27652107	29,8602694
929	45,37966697	92,9011809
981	55,35158165	84,6615992
988	42,4572432	96,4639819
992	50,35412714	90,1798904
1037	40,42412482	96,6708134
1049	94,69304828	5,65878448
1212	82,18965728	23,6074254

TABLE 4.4: Tableau relatif à la moyenne et à l'étendue des charges des agents de diagnostic d'un déploiement distribué de la fonction de diagnostic

L'exploitation des courbes permet de mettre en avant les valeurs, en termes de nombre de données traitées, pour lesquelles le déploiement distribué de la fonction

de diagnostic a été plus performante que son déploiement centralisé. En effet, lors du traitement de 1049 données, la vitesse de convergence du déploiement distribué est de 151 secondes et de 216 secondes pour le déploiement centralisé, soit une différence de 65 secondes. Pour 1037 données traitées, la vitesse est de 159 secondes pour le distribué et de 171 secondes, différence : 12 secondes seulement. L'étendue étant supérieure à la moyenne dans ce cas là, l'apport du distribué n'est pas optimal par rapport au centralisé. L'apport de l'approche distribuée est donc bien meilleur lorsque la charge de chacun des agents de diagnostic qui la compose est proche de 100% sous l'hypothèse que le nombre de données à traiter soit suffisamment élevé (supérieur à 450).

4.6. Conclusion

Dans ce chapitre nous avons présenté les principes distribués de diagnostic et de pronostic issus des principes proposés dans le chapitre 3. La plateforme, configurable par les connaissances présentées dans le chapitre 2, que nous avons développé afin d'évaluer et comparer le déploiement distribué et le déploiement centralisé du principe de diagnostic a été présentée. Le cas d'étude servant à cette évaluation et comparaison a également été décrit. Il a permis la validation du principe de diagnostic pour un déploiement centralisé et pour un déploiement distribué. Un artifice sur la génération des symptômes nous a permis d'évaluer les vitesses de convergence des deux déploiements. Les résultats des simulations effectuées montrent que pour un grand nombre de symptômes à traiter presque simultanément le déploiement distribué avait une vitesse de convergence inférieure à celle du déploiement centralisé. Cependant, cette vitesse semble dépendre de la répartition du nombre de symptômes que doit traiter quasiment simultanément chacun des agents du déploiement distribué et donc de leur charge de calcul.

Nous avons constaté que l'écart entre les vitesses de convergence des deux déploiements variait très peu au fur et à mesure que le nombre de données traitées augmentait. Cela peut être dû au fait que tous les éléments de l'architecture sont implémentés sur la même plateforme de calcul. Nous supposons que cela peut correspondre, pour un grand nombre de données traitées, a effectué un déploiement centralisé. En effet, cette implémentation mono plateforme ne permet pas de bénéficier du parallélisme des traitements réalisés par les agents du déploiement distribué implémentés sur des plateformes différentes.

Il est possible de découper la connaissance en plusieurs parties dans le cas d'un déploiement distribué des fonctions de diagnostic et de pronostic. Ce découpage

peut être effectué selon la nature de la connaissance (structurelle, topologique ou comportementale). Chacune de ces bases pouvant ensuite être hébergée sur des plateformes différentes. Le découpage peut également être fonction du champ d'action qui a été donné aux différents agents de l'architecture. Ce champ d'action étant, dans notre proposition, relatif à la fonction surveillée, la connaissance devra au minimum comporter :

- la connaissance fonctionnelle pour la fonction,
- les connaissances structurelles et topologiques propres aux composants de la fonction et aux composant d'autres fonctions directement dépendant de composants de cette fonction mais aussi des composants de cette fonction qui dépendent directement de composants d'autres fonctions,
- la connaissance comportementale associant les symptômes des composants de la fonction à leurs modes de défaillance.

Il est aussi possible de répartir le « blackboard ». L'intérêt réside dans le fait que chacun des agents disposera de son propre espace de stockage. Il est alors possible d'envisager une implémentation où une plateforme de calcul serait dédiée à une fonction du STCGD. Cette plateforme hébergerait alors, l'agent de diagnostic, l'agent de pronostic, la base de connaissance et, soit un « blackboard » par agent, soit un « blackboard » commun regroupant l'ensemble des éléments contribuant à l'évaluation de l'état de santé de la fonction et de ces composants. D'autres implémentations peuvent également être considérées.

Conclusion générale

Nous avons proposé une architecture distribuée pour le diagnostic et le pronostic de systèmes techniques complexes de grande dimension permettant d'en définir l'état de santé. Un STCGD a été défini comme un système mettant en œuvre différents systèmes de multiples fournisseurs interagissant entre eux dans un but commun et devant répondre, la plupart du temps, à des critères de performances, de fiabilité et de sécurité prédéfinis. Chaque système est réalisé par un certain nombre de sous-systèmes eux-même implémentant des fonctions. Ces fonctions sont également constituées d'interconnexions de composants remplaçables et/ou réparables. Durant son utilisation, le système est sujet à diverses défaillances ayant un impact sur son mode de fonctionnement contraignant leurs utilisateurs à s'adapter à ces nouvelles conditions de conduite d'un système en mode dégradé. Un système doit être considéré selon deux points de vue : concepteurs et utilisateurs au sens large du terme. En effet, il existe deux types d'acteurs agissant sur le système : les opérateurs et les mainteneurs. Chacun d'eux a des besoins différents concernant le résultat que leur fournira le système de surveillance. L'opérateur voudra connaître les fonctionnalités indisponibles de son système car cela aura des conséquences opérationnelles durant la mission à assurer. Pour le mainteneur, les composants seront sa seule priorité. Ainsi, afin d'aider les différents acteurs du système et les aider dans la conduite ou le maintien en condition opérationnelle, il faut définir une fonction de diagnostic fournissant des informations non erronées dans un temps raisonnable. Nous avons proposé une fonction de pronostic pour satisfaire la préoccupation des exploitants relative à la capacité de leurs systèmes à être engagés dans des conditions de fiabilité acceptables. Un autre intérêt d'une fonction pronostic est de pouvoir être proactive en anticipant les futures défaillances et de permettre l'établissement d'un planning prévisionnel de maintenance optimisé pour assurer un engagement optimal des composants du système. Pour aider à l'évaluation de l'état de santé des STCGD, il est nécessaire d'avoir une certaine connaissance du système que l'on veut engager et/ou maintenir. Différentes méthodes, desquelles la

connaissance peut être extraite, peuvent être utilisées à l'élaboration d'une évaluation de l'état de santé. Celles-ci sont issues principalement d'études menées durant la phase de conception du système. A cela peut s'ajouter le retour d'expérience qui permettra d'affiner la connaissance a priori que l'on a du système. Notre approche du diagnostic est basée sur quatre types de connaissances : fonctionnelle, structurelle, comportementale et topologique. L'analyse fonctionnelle recense les fonctions du système physique qui possède des indicateurs pertinents et significatifs. Ces fonctions correspondent à l'analyse du système physique, aux données de conception, à l'analyse experte. L'analyse structurelle s'applique aux fonctions identifiées en recensant l'ensemble des composants et les connexions physiques entre eux. L'analyse comportementale permet de recenser les variables et leurs relations (relation de causes à effet, de causalité). L'analyse topologique permet de déterminer les proximités entre les objets pouvant entraîner une défaillance indirecte. Différents types d'architectures permettant de déployer ces fonctions de surveillance peuvent être considérés lors de la conception d'un système. Cependant, chacune de ces architectures présente ses propres avantages et inconvénients. Dans une architecture centralisée de diagnostic un seul et unique organe est chargé d'établir un diagnostic, posant évidemment des problèmes de fiabilité. Pour pallier le problème de charge du calculateur principal d'une architecture centralisée et ainsi pouvoir surveiller des systèmes fournissant des quantités d'informations supérieures, les architectures décentralisées ont été étudiées. Elles permettent de diminuer la charge des données à traiter en la distribuant sur des agents de diagnostic locaux. Ceux-ci envoient ensuite leurs résultats à un agent de diagnostic global qui assure le diagnostic de l'ensemble du système. Mettre en place une approche distribuée permet de pallier le problème de fiabilité du système. En effet, aucun organe n'est hiérarchiquement en charge d'établir un diagnostic global du système, ce qui implique qu'un diagnostic global existe toujours à moins que tous les agents locaux tombent en panne. Chacun des agents locaux de diagnostic doit communiquer avec les autres afin de tendre localement et globalement vers un diagnostic cohérent. La spécification et la formalisation des différentes architectures (centralisées, décentralisées et distribuées), des méthodes implémentées et des critères de performances ont été réalisées. La quantification des critères de performance de différentes architectures centralisées, décentralisées et distribuées pour le diagnostic et le pronostic de systèmes a été entreprise. L'étude et les évaluations menées dans ce mémoire ont montré qu'un déploiement distribué de la fonction de diagnostic était plus performante en terme de vitesse de convergence qu'un déploiement centralisé lorsque le nombre de données à traiter dépasse les

350/400 données (symptômes et Faulty_LRU_Message confondus). En effet, c'est à partir de cette limite que les coûts, en termes de temps, de la communication entre les agents d'un déploiement distribué deviennent négligeables par rapport au temps de traitement. En deça de cette limite, le déploiement centralisé est plus rapide car demandant moins de communication.

Perspectives

Une hypothèse simplificatrice de ce mémoire porte sur la connaissance exacte de la date d'apparition des symptômes. Dans la réalité, certains systèmes, soumis à une contrainte de poids et à l'utilisation de matériels certifiés aux performances restreintes, ne peuvent pas se permettre d'installer une fonction de surveillance sur tous les équipements. Dès lors, une fonction de surveillance devra surveiller plusieurs fonctions en s'octroyant des fenêtres temporelles dédiées à son activité pour chacunes d'entre elles. Ainsi, nous n'aurions plus à traiter un symptôme avec un attribut de type date d'apparition, mais avec 2 dates d'apparition : l'une définissant le dernier instant où le symptôme n'était pas encore présent et l'autre, le premier instant où le symptôme est apparu. Nous aurons donc à traiter un intervalle dans lequel le symptôme est apparu. La fonction de diagnostic devra donc être modifiée pour prendre en compte cette nouvelle contrainte. Les résultats de diagnostic comporteront donc plusieurs alternatives accompagnées d'une probabilité de confiance. De plus, les algorithmes de diagnostic devront être redéfinis afin de permettre un déploiement décentralisé de la fonction et être intégrés aux résultats déjà obtenus. La fonction pronostic quant à elle, même si celle-ci n'était pas notre priorité, a été définie et pourra être déployée et faire aussi l'objet d'une évaluation et d'une comparaison. La fonction de pronostic proposée est monotone. Cette fonction n'est pas encore très répandue dans les systèmes développés actuellement et fait toujours l'objet de travaux de recherche comme nous l'avons montré dans le chapitre 1. Une évolution de l'architecture porterait aussi sur la possibilité des agents de diagnostic (et de pronostic) à se surveiller les uns les autres et ainsi d'assurer le diagnostic du système d'évaluation d'état de santé. Une piste de travail porte sur la capacité des agents à traiter le contenu du message qui leur est retourné lors de l'envoi d'une donnée. En effet, lorsqu'une donnée est envoyée par un client, le réseau retourne toujours un message au client lui signifiant si le message a bien été transmis. Dès lors, une fois la défaillance d'un des agents identifiée, un mécanisme devra s'assurer que les données devant être traitées par

cet agent le soit par d'autres. Cela permettrait ainsi d'assurer la continuité de la tâche d'évaluation d'état de santé sans perte d'informations afin de fournir une qualité de service augmentée.

Bibliographie

M. Albert, T. Längle et H. Wörn : Development tool for distributed monitoring and diagnosis systems. *In Proceedings of Thirteenth International Workshop on Principles of Diagnosis*, p. 158–164, 2002.

H. Alex, M. Kumar et B. Shirazi : Midfusion : middleware for information fusion in sensor network applications. *In Information Fusion Journal*, p. 617–622, 14-17 Dec. 2004.

L. Ardissono, L. Console, G. Goy, A. and. Petrone, C. Picardi, M. Segnan et D. Theseider Dupré : Advanced fault analysis in web service composition. *In WWW '05 : Special interest tracks and posters of the 14th international conference on World Wide Web*, p. 1090–1091, New York, NY, USA, 2005. ACM. ISBN 1-59593-051-5.

V. Ariton, V. Palade et F. Postolache : Combined deep and shallow knowledge in a unified model for diagnosis by abduction. *Euroeconomica*, 1(20):33 – 42, 2008.

V. Ariton : Deep and shallow knowledge in fault diagnosis. *In Knowledge-Based Intelligent Information and Engineering Systems*, vol. 2773 de *Lecture Notes in Computer Science*, p. 748–755. Springer Berlin / Heidelberg, 2003.

N. Audsley et A. Grigg : Timing analysis of the arinc 629 databus for real-time applications. *Microprocessors and Microsystems*, 21:55–61(7), July 1997.

M. Azam, F. Tu et K. R. Pattipati : Condition-based predictive maintenance of industrial power systems. *Component and Systems Diagnostics, Prognostics, and Health Management II*, 4733:133–144, 2002.

C. Barnhart et R. Ziemer : Topological analysis of satellite-based distributed sensor networks. *Systems, Man and Cybernetics, IEEE Transactions on*, 21, Issue 5:1060 – 1070, Sep/Oct 1991.

M. Bengtsson, E. Olsson, P. Funk et M. Jackson : Technical design of condition based maintenance system a case study using sound analysis and case-based reasoning. *In Maintenance and Reliability Conference Proceedings*

of the 8th Congress, May 2004.

S. Berger : Arinc 629 digital communication system - application on the 777 and beyond. *Microprocessors and Microsystems*, 20, Number 8:463–471(9), April 1997.

H. Bersini : *La programmation orientée objet.* Eyrolles, 2008.

J. Biteus : Distributed diagnosis and simulation based residual generators. Rap. tech., Dept. of Electrical Engineering, 2005. LiU-TEK-LIC-2005 :31, Thesis No. 1176.

J. Biteus, M. Jensen et M. Nyberg : Distributed diagnosis for embedded systems in automotive vehicles. IFAC World Congress, Prague, Czech Republic, 2005.

J. Biteus, M. Nyberg et E. Frisk : An algorithm for computing the diagnoses with minimal cardinality in a distributed system. *Engineering Applications of Artificial Intelligence*, 21(Issue 2):269–276, 2008.

R. K. Boel et J. H. van Schuppen : Decentralized failure diagnosis for discrete-event systems with costly communication between diagnosers. *In WODES '02 : Proceedings of the Sixth International Workshop on Discrete Event Systems (WODES'02)*, p. 175, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1683-1.

W. Bolton : *Les automates programmables industriels.* Technique et Ingénierie, 2010.

C. Byington, M. Roemer, M. Watson et T. Galie : Prognostic enhancements to gas turbine diagnostic systems. *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, 7:3247–3255, March 2003.

C. S. Byington, M. J. Roemer et T. Galie : Prognostic enhancements to diagnostic systems for improved condition-based maintenance. *Aerospace Conference Proceedings*, 6:2815–2824, 2002.

C. Byington, P. Kalgren et B. Donovan : Portable diagnostic reasoning for improved avionics maintenance and information capture & continuity. *AUTOTESTCON 2004. Proceedings*, p. 518–524, 20-23 Sept. 2004a. ISSN 1088-7725.

C. Byington, M. Watson, D. Edwards et P. Stoelting : A model-based approach to prognostics and health management for flight control actuators. *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, 6:3551–3562, 6-13 March 2004b. ISSN 1095-323X.

S. Cavalieri, A. Di Stefano et O. Mirabella : Centralized versus distributed protocols for fieldbus applications. *In Industrial Electronics, Control, and Instrumentation*, vol. 2, p. 1580–1585, Nov 1995.

G. Cena, L. Durante et A. Valenzano : Standard field bus networks for industrial applications. *Computer Standards & Interfaces*, 17(2):155 – 167, 1995. ISSN 0920-5489.

Y. Chen et M. Lee : Neural networks-based scheme for system failure detection and diagnosis. *Mathematics and computers in simulation*, 58:101–109, 2002. Issue 2.

L. Chiang, E. Russel et R. Braatz : *Fault detection and diagnosis in industrial systems*. SpringerEd. Springer-Verlag, 2001.

L. Chittaro, G. Guida, C. Tasso et E. Toppano : Functional and teleological knowledge in the multimodeling approach for reasoning about physical systems : a case study in diagnosis. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(issue 6):1718–1751, Nov/Dec 1993.

CIAME : *Réseaux de terrain : description et critères de choix*. 1999.

CIAME : *Réseaux de terrain : Critères de sûreté de fonctionnement*. Lavoisier, 2009.

M. Combacau, P. Berruet, E. Zamai, P. Charbonnaud et A. Khatab : Supervision and monitoring of production systems. *In MCPL2000*, Grenoble, France, 5 to 8 july 2000.

L. Console, C. Picardi et D. Theseider Dupré : A framework for decentralized qualitative model-based diagnosis. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, Hyderabad , India, January 6-12 2007.

R. Debouk, S. Lafortune et D. Teneketzis : On the effect of communication delays in failure diagnosis of decentralized discrete event systems. *Discrete Event Dynamic Systems*, 13(3):263–289, 2003. ISSN 0924-6703.

L. Debrauwer et F. Van Der Heyde : *UML 2 : Initiation, exemples et exercices corrigés*. Editions ENI, 2008.

X. Desforges et B. Archimède : Multi-agent framework based on smart sensors/actuators for machine tools control and monitoring. *Engineering Applications of Artificial Intelligence*, 19(6):641–655, september 2006.

M. Dievart et P. Charbonnaud : Decentralized versus distributed diagnosis. French German Workshop on Diagnosis, Airbus, Toulouse, France, November 2007.

M. Dievart, P. Charbonnaud et X. Desforges : Applicative architecture for embedded distributed technical diagnosis. *In 7th Workshop on Advanced Control and Diagnosis*, 2009.

M. Dievart, P. Charbonnaud et X. Desforges : Distributed embedded health

assessment of networked instruments. *In Large Scale Systems : Theory and Applications (12th IFAC-LSS Symposium)*, 2010a.

M. Dievart, P. Charbonnaud et X. Desforges : An embedded distributed tool for transportation systems health assessment. *In Embedded Real Time Software and Systems (ERTSS)*, 2010b.

M. Dievart, X. Desforges et P. Charbonnaud : Architecture distribuée de diagnostic embarqué des systèmes complexes. *In 9ième Congrès EDSYS*, 2008.

O. Dragomir, R. Gouriveau, N. Zerhouni et F. Dragomir : Framework for a distributed and hybrid prognostic system. *In 4th IFAC Conference on Management and Control of Production and Logistics*. Sibiu, Romania, 2007.

S. Engel, B. Gilmartin, K. Bongort et A. Hess : Prognostics, the real issues involved with predicting life remaining. *Aerospace Conference Proceedings, 2000 IEEE*, 6:457–469, mars 2000.

M. C. Garcia, M. A. Sanz-Bobi et J. del Pico : Simap : intelligent system for predictive maintenance application to the health condition monitoring of a windturbine gearbox. *Comput. Ind.*, 57(6):552–568, 2006. ISSN 0166-3615.

C. Garcia-Beltran : *Outils pour l´aide á la supervision de procédés dans une architecture multiagent*. Thèse de doctorat, Institut National Polytechnique de Grenoble, 2004.

. Gardarin et O. Gardarin : *Le Client-Serveur*. Eyrolles, 1996.

A. K. Garga, K. T. McClintic, R. L. Campbell, C.-C. Yang, M. S. Lebold, T. Hay et C. Byington : Hybrid reasoning for prognostic learning in cbm systems. *Aerospace Conference, 2001, IEEE Proceedings*, 6:2957–2969, 2001.

S. Glavaski et M. Elgersma : Active aircraft fault detection and isolation. *In AUTOTESTCON Proceedings, 2001. IEEE Systems Readiness Technology Conference*, p. 692–705, 08/20 to 08/23 2001.

J. Grieu : *Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques*. Thèse de doctorat, Institut National Polytechnique de Toulouse, 2004.

P. Halder, C. S.K. et S. Mukhopadhyay : On-line sensor fault detection, isolation and accomodation in tactical aerospace vehicle. *TENCON 2004. 2004 IEEE Region 10 Conference*, 4:684–686, november 2004.

D. Hallgren et H. Skog : *Distributed fault diagnosis for networked embedded systems*. Thèse de doctorat, Linkoping University, 2005.

D. Haverkamp et R. Richards : Towards safety critical middleware for avionics applications. *In Local Computer Networks, 2002. Proceedings. LCN 2002. 27th Annual IEEE Conference on*, p. 716– 722, 6-8 November 2002.

S. Heath : *Embedded systems design*. Num. ISBN 9780750655460 de EDN series for design engineers. Newnes, 2003.

M. Hedley, M. Johnson, C. Lewis, D. Carpenter, H. Lovatt et D. Price : Smart sensor network for space vehicle monitoring. *In Proceedings of the International Signal Processing Conference*, March 2003.

W. B. Heinzelman, A. L. Murphy, H. S. Carvalho et M. A. Perillo : Middleware to support sensor network applications. *IEEE Network*, 18(1):6–14, 2004.

K. Henricksen et R. Robinson : A survey of middleware for sensor networks : state-of-the-art and future directions. *In MidSens '06 : Proceedings of the international workshop on Middleware for sensor networks*, p. 60–65, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-424-3.

A. M. Ibrahim : Embedded systems : An overview. *In Fuzzy Logic for Embedded Systems Applications*, p. 1 – 21. Newnes, Burlington, 2004. ISBN 978-0-75-067605-2.

R. Isermann : Supervision, fault-detection and fault-diagnosis methods-an introduction. *Control engineering practice*, 5:639–652, may 1997. Issue 5.

A. Jardine, D. Lin et D. Banjevic : A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20:1483–1510, october 2006.

N. Jennings et M. Wooldridge : Applying agent technology. *Applied Artificial Intelligence*, 9(4):357–369, 1995.

R. Kothamasu, S. H. Huang et W. H. VerDuin : System health monitoring and prognostics - a review of current paradigms and practices. *The International Journal of Advanced Manufacturing Technology*, 28(9):1012–1024, april 2006.

M. Lebold, K. Reichard, C. Byington et R. Orsagh : Osa-cbm architecture development with emphasis on xml implementations. *In Maintenance and Reliability Conference (MARCON)*, 2002.

M. Lebold et M. Thurston : Open standards for condition-based maintenance and prognostics systems. *In 5th annual maintenance and reliability conference (MARCON 2001)*, 2001.

M. Lebold, K. McClintic, R. Campbell, C. Byington et K. Maynard : Review of vibration analysis methods for gearbox diagnostics and prognostics. *In Proceedings of the 54th Meeting of the Society for Machinery Failure Prevention Technology*, p. 623–634, Virginia Beach, VA, 1-4 May 2000.

Y. Li, L. Chun et A. N. Y. Ching : An agent-based platform for web-enabled equipment predictive maintenance. *In IAT '05 : Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Techno-*

logy, p. 132–135, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2416-8.

N. Limnios : *Arbres de défaillances*. Hermes, 2005.

W. Liu : An on-line expert system-based fault-tolerant control system. *Expert systems with applications*, 11:59–64, 1996. Issue 1.

J. Luo, M. Namburu, K. Pattipati, L. Qiao, M. Kawamoto et S. Chigusa : Model-based prognostic techniques. *In AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference. Proceedings*, p. 330–340, september 2003.

A. Mathur, K. Cavanaugh, K. Pattipati, P. Willett et T. Galie : Reasoning and modeling systems in diagnosis and prognosis. *In* P. K. Willett et T. Kirubarajan, éds : *Proc. SPIE Vol. 4389, p. 194-203, Component and Systems Diagnostics, Prognosis, and Health Management, Peter K. Willett; Thiagalingam Kirubarajan; Eds.*, vol. 4389 de *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, p. 194–203, juil. 2001.

J. Meinadier : *Ingénierie et intégration des systèmes*. Hermes Sciences Publication, 1998.

B. Meyer et P. Jouvelot : *Conception et programmation orientées objet*. Eyrolles, 2008.

A. Mifdaoui : *Spécification et validation d´un réseau de communication de type Ethernet Commuté pour systèmes avioniques militaires de nouvelles générations*. Thèse de doctorat, Institut National Polytechnique de Toulouse, 2007.

A. Muller : *Contribution á la maintenance prévisionnelle des systèmes de production par la formalisation d'un processus de pronostic*. Thèse de doctorat, Université Henri Poincaré, Nancy I, 2005.

V. Murthy et E. Krishnamurthy : Multiset of agents in a network for simulation of complex systems. *In* K. e. a. Springer Verlag, éd. : *Recent advances in Nonlinear Dynamics and synchronization, (NDS-1) -Theory and applications*, 2009.

H. Niemann, N. K. Poulsen et M. A. B. Baekgaard : A multi-model approach for system diagnosis. *In American Control Conference, 2007. ACC '07*, New York, NY, USA, 9-13 July 2007.

Y. Papadopoulos : Model-based system monitoring and diagnosis of failures using statecharts and fault trees. *Reliability and security of industrial computer systems*, 81:325–341, 2003. Issue 3.

R. Patton : Robustness in model-based fault diagnosis : the 1995 situation. *A. Rev. Control*, 21:203–123, 1997.

R. Patton, P. Franck et R. Clark : *Fault diagnosis in dynamic systems :*

theory and applications. Englewood Cliffs, N.J. : Prentice Hall, 1989.

Y. Pencolé et M.-O. Cordier : A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks. *Artif. Intell.*, 164(1-2):121–170, 2005. ISSN 0004-3702.

G. Provan : An open systems architecture for prognostic inference during condition-based monitoring. *Aerospace Conference*, 7:3157–3164, 2003.

A. Puder, K. Romer et F. Pilhofer : *Distributed Systems Architecture : A Middleware Approach*. Elsevier Science Inc., 2005.

W. Qiu et R. Kumar : Decentralized failure diagnosis of discrete event systems. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS - PART A : SYSTEMS AND HUMANS*, 36(2):384–395, March 2006.

X. Rebeuf, N. Blanc, F. Charpillet, D. Chevé, A. Dutech, C. Lang, L. Pélissier et J. Thomesse : Proteus, des web services pour les systèmes de maintenance. *In Nouvelles Technologie de la Répartition - NOTERE'04*, 2004.

R. Reiter : A theory of diagnosis from first principles. *In Readings in model-based diagnosis*, p. 29–48, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc. ISBN 1-55860-249-6.

P. Ribot : *Vers l´intégration diagnostic/pronostic pour la maintenance des systèmes complexes*. Thèse de doctorat, Université Toulouse 3 Paul Sabatier, 2009.

M. Robert, M. Porte et M. Marchandiaux : *Capteurs intelligents et méthodologie d'évaluation*. 1993.

M. Roemer, C. Byington, G. J. Kacprzynski et G. Vachtsevanos : An overview of selected prognostic technologies with reference to an integrated phm architecture. Rap. tech., Impakt Technologies, 2007.

A. Sadovykh : *Concept innovateur d´un middleware pour la supervision de systèmes complexes*. Thèse de doctorat, Université Paris VI Pierre et Marie Curie, 2005.

B. Saha, S. Saha et K. Goebel : A distributed prognostic health management architecture. *In Proc. of the Conference of the Society for Machinery Failure Prevention Technology*, 2009.

G. C. Sakellaropoulos et G. C. Nikiforidis : Prognostic performance of two expert systems based on bayesian belief networks. *Decis. Support Syst.*, 27 (4):431–442, 2000. ISSN 0167-9236.

H. Saranga et J. Knezevic : Reliability prediction for condition-based maintained systems. *Reliability Engineering and System Safety*, 71(6):219–224, 2001.

R. E. Schantz et D. C. Schmidt : Research advances in middleware for distributed systems. *In Proceedings of the IFIP 17th World Computer Congress - TC6*

Stream on Communication Systems : The State of the Art, p. 1–36, Deventer, The Netherlands, The Netherlands, 2002. Kluwer, B.V. ISBN 1-4020-7168-X.

N. Simoni : *Des réseaux intelligents á la nouvelle génération de services.* Hermès Lavoisier, 2007.

C. G. Siontorou, F. A. Batzias et V. Tsakiri : A knowledge-based approach to online fault diagnosis of fet biosensors. *Instrumentation and Measurement, IEEE Transactions on*, 59(9):2345 – 2364, 2010.

M. Staroswiecki et M. Bayart : *Actionneurs intelligents.* 1994.

D. Stipanicev et J. Marasovic : Networked embedded greenhouse monitoring and control. *Control Applications, 2003. CCA 2003. Proceedings of 2003 IEEE Conference on*, 2:1350–1355, june 2003.

S. Takai et R. Kumar : Distributed prognosis of discrete event systems under bounded-delay communications. *IEEE Conference on Decision and Control and 28th Chinese COntrol Conference*, 1:1235–1240, 2009.

S. Touaf : *Diagnostic logique des systèmes complexes dynamiques dans un contexte multi-agent.* Thèse de doctorat, L´université Joseph Fourier - Grenoble 1, 2005.

G. Vachtsevanos et P. Wang : Fault prognosis using dynamic wavelet neural networks. *In AUTOTESTCON Proceedings, 2001. IEEE Systems Readiness Technology Conference*, p. 857–870, 2001.

G. Vachtsevanos, F. L. Lewis, M. Roemer, A. Hess et B. Wu : *Intelligent Fault Diagnosis and Prognosis for Engineering Systems.* John Wiley & Sons, 2006.

A. Villemeur : *Sûreté de fonctionnement des systèmes industriels : Fiabilité-Facteurs humains-Informatisation.* Eyrolles, 1988.

C. Wilkinson, D. Humphrey, B. Vermeire et J. Houston : Prognostic and health management for avionics. *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, 5:3435–3447, march 2004.

H. Worn, T. Langle, M. Albert, A. Kazi, A. Brighenti, S. Revuelta Seijo, C. Senior, M. A. Sanz Bobi et J. Villar Collado : Diamond : distributed multi-agent architecture for monitoring and diagnosis. *Production Planning and Control*, 15:189–200(12), 2004.

X. Wu et G. Campion : Fault detection and isolation of systems with slowly varying parameters- simulation with a simplified aircraft turbo engine model. *Mechanical systems and signal processing*, 18:353–366, march 2004. Issue 2.

J. Yan, J. Lee et M. Koç : Predictive algorithm for machine degradation detection using logistic regression. *In Managing innovative manufacturing,*

"e-manufacturing abd e-business integration" (MIM'2002), 2002.

R. Zemouri : *Contribution á la surveillance des systèmes de production á l'aide de réseaux de neurones dynamiques : Application á la e-maintenance.* Thèse de doctorat, Université de Franche-Comté, 2003.

Y. Zennir : *Apprentissage par renforcement et systèmes distribués : Application á l'apprentissage de la marche d'un robot hexapode.* Thèse de doctorat, INSA de Lyon, 2004.

A. Zolghadri : Early warning and prediction of flight parameter abnormalities for improved system safety assessment. *Reliability Engineering and System Safety*, 76(1):19–27(9), April 2002.

G. Zwingelstein : *Diagnostic des défaillances.* Traité de nouvelles technologies, série diagnostic et maintenance, 1995.