HAL
open science

Test et diagnostic en ligne dans les systèmes embarqués

Oum-El-Kheir Aktouf

To cite this version:

Oum-El-Kheir Aktouf. Test et diagnostic en ligne dans les systèmes embarqués. Informatique [cs]. Grenoble 1 UGA - Université Grenoble Alpe, 2014. tel-01731434

HAL Id: tel-01731434

https://hal.science/tel-01731434

Submitted on 14 Mar 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Mémoire présenté par

Oum-El-Kheir Aktouf

En vue de l'obtention de

L'Habilitation à Diriger des Recherches

Université de Grenoble

Test et diagnostic en ligne dans les systèmes embarqués

Soutenu le 11 juillet 2014 devant le jury composé de :

Mme Catherine Dubois	Professeur à l'ENSIIE	Rapporteur
M. Daniel Hagimont	Professeur à l'ENSEEIHT	Rapporteur
M. Matteo Sonza Reorda	Professeur au Politecnico di Torino	Rapporteur
M. Mohamed Akil	Professeur à l'ESIEE	Examinateur
M. Yves Ledru	Professeur à l'Université Grenoble I	Examinateur
M. Ioannis Parissis	Professeur à Grenoble INP	Examinateur

SOMMAIRE

Sommaire 3
Remerciements 5
Introduction 7
Chapitre 1 Test et Diagnostic en ligne de composants logiciels embarqués 9
Chapitre 2 Test et diagnostic en ligne des systèmes RFID 33
Conclusion générale 67
Bibliographie 69
Table des Matières 79
Annexes 81

REMERCIEMENTS

Je tiens à exprimer mes plus vifs remerciements aux membres du jury pour le temps consacré à ce mémoire et à la soutenance :

Mme Catherine Dubois, Professeur à l'ENSIIE ; M. Daniel Hagimont, Professeur à l'ENSEEIHT ; M. Matteo Sonza Reorda, Professeur au Politecnico di Torino ; M. Mohamed Akil, Professeur à l'ESIEE ; M. Yves Ledru, Professeur à l'Université Grenoble I et M. Ioannis Parissis, Professeur à Grenoble INP.

Mme Dubois, M. Hagimont et M. Sonza Reorda ont eu la charge de rapporter sur le mémoire, je les en remercie chaleureusement.

Ma plus profonde gratitude s'adresse à M. Michel Dang, Professeur Honoraire des Universités, pour les conseils qu'il n'a eus de cesse de me prodiguer dès mon recrutement à l'Esisar. Grâce à son aide, j'ai pu relever le défi de mener des activités de recherche dans un jeune laboratoire, le LCIS, et de participer modestement à son développement.

L'une des particularités de ce laboratoire est de regrouper des enseignants-chercheurs de diverses disciplines : physique, automatique, électronique, informatique, *etc*. Je remercie les différents directeurs qui se sont succédés : Smail Tedjini, Didier Georges, Chantal Robach, Eduardo Mendès et Michel Occello, et qui ont œuvré pour en faire une unité cohérente et accueillante tant pour les doctorants que pour les enseignants-chercheurs accueillis au fil des années.

Les travaux de recherche présentés dans ce mémoire sont collectifs. A ce titre, je remercie les collègues avec qui j'ai pu avoir des échanges intéressants et riches et des co-encadrements de thèses : Vincent, Ioannis, David ainsi que les Doctorants avec qui j'ai partagé l'aventure d'une thèse : Thi Quynh, Gilles, Dima et Rafik.

Aimer son métier, ce qui est mon cas, est une réelle chance, mais pouvoir en plus l'exercer dans un environnement agréable et y côtoyer chaque jour des collègues devenus des amis, est simplement une bénédiction. Je ne pourrai pas citer tout le monde, mais que tous les personnels du LCIS et de l'Esisar soient assurés de mon amitié. Je remercie en particulier les directeurs successifs de l'Esisar qui ont œuvré à en faire une école dynamique où l'environnement de travail est agréable : Michel Dang, Smail Tedjini, Chantal Robach et Nadine Guillemot.

Pour finir, je souhaiterais remercier les personnes qui me sont les plus chères et leur dédier ce travail : mes parents Houria et Ahcène, ma sœur Salima et mon frère Chérif, mes enfants Yanis et Sami et mon époux Chouki. C'est grâce à leur aide, leur soutien et leurs encouragements que ce travail a pu aboutir.

Merci à tous.

Valence, le 7 juillet 2014, O. Aktouf

INTRODUCTION

Ce mémoire décrit les travaux menés sur la sûreté de fonctionnement des systèmes embarqués au sein de l'équipe CTSYS (Conception et Test des Systèmes Embarqués) du laboratoire LCIS (Laboratoire de Conception et d'Intégration des Systèmes). Ce laboratoire, fondé en 1996, se caractérise par une forte pluridisciplinarité puisque les recherches qui y sont menées concernent tant la physique, l'automatique, l'électronique que l'informatique. Le thème fédérateur de ces différents axes concerne les « systèmes embarqués communicants ». Ce choix a été motivé par l'importance, toujours grandissante, des systèmes embarqués, entraînant ainsi de nouveaux besoins de recherche.

L'évolution récente de l'informatique a donné naissance à des systèmes de plus en plus sophistiqués, intégrant différents types de ressources (capteur, téléphone mobile, lecteur et étiquette RFID, logiciel embarqué, *etc*.) Ces systèmes se caractérisent à des degrés divers par leur répartition, leur environnement variable et souvent hostile (perturbations électromagnétiques, milieu non conducteur bloquant la communication, *etc*.) et les contraintes de ressources et d'énergie. En outre, en raison de la nature de certaines applications (surveillance et suivi à distance de malades, détection de substances toxiques dans des environnements industriels ou autres, inventaire de grands stocks d'articles, *etc*.) il est de plus en plus difficile et coûteux d'avoir recours à l'intervention humaine en cas de dysfonctionnement.

La problématique de la sûreté de fonctionnement de tels systèmes devient alors cruciale, et doit tenir compte de toutes ces contraintes tout en limitant le coût induit par les mécanismes mis en œuvre. Or, les solutions classiques de redondance structurelle peuvent ne pas être adaptées car la réplication des équipements communicants dans de tels systèmes peut augmenter leur propension aux pannes, en engendrant par exemple des interférences supplémentaires. Ceci a motivé l'orientation de nos travaux vers des solutions fondées principalement sur l'utilisation de *tests* et de *diagnostic en ligne*.

En outre, la nature embarquée des différents éléments nécessite, pour une appréhension globale du système étudié, la prise en compte des composants matériels, logiciels et de communication (réseaux) qui forment le système étudié. En effet, les interactions entre ces différents types de composants peuvent influer sur la qualité globale du système.

Dans l'optique de réduire les coûts en ressources liés à la sûreté de fonctionnement, mais également dans le but de proposer des solutions de tolérance aux fautes « autonomes » répondant aux contraintes des nouvelles applications informatiques, nous nous intéressons en particulier à des approches basées sur le diagnostic de système. Ce type de diagnostic utilise les ressources propres d'une application ou d'un système pour assurer la détection et la localisation des éléments défaillants, au moyen de *tests inter-composants* ou de *test en ligne*, conférant ainsi une certaine autonomie aux systèmes et applications. De nombreux verrous scientifiques et techniques doivent être levés, en particulier la faisabilité des tests inter-composants en ligne, la définition de la nature des fautes traitées, l'analyse des résultats des tests pour en déduire l'état du système, la reconfiguration, le passage à l'échelle, *etc*.

Toutefois, il ressort de ces différents travaux qu'un composant clé, le *middleware* ou l'*intergiciel* peut servir de support à la solution de sûreté de fonctionnement choisie, en intégrant la gestion des tests inter-composants et le diagnostic. Son positionnement en tant qu'interface entre le « monde applicatif » et le « monde physique » motive également une telle orientation de travail.

Les domaines d'application de ces recherches concernent les applications à base de composants logiciels (projet DISCO), les systèmes RFID (projet ANR SafeRFID) et les applications basées sur les réseaux de capteurs (projets ARTeco et CEDRE). Du point de vue régional, ils relèvent du projet Semba (Systèmes Embarqués) du cluster ISLE (Informatique, Signal, Logiciel Embarqué) ainsi que de l'ARC6 (Communauté de Recherches Académiques Rhône-Alpes). A l'international, ils nous ont permis de participer au projet européen (Réseau d'Excellence) EURNEX sur la sûreté dans le domaine ferroviaire.

Ce rapport a pour objectif de décrire l'ensemble des travaux menés, leur état d'avancement et les principaux résultats obtenus. Il s'articule autour de deux chapitres principaux qui couvrent différents aspects des systèmes embarqués, concernant respectivement :

- les composants logiciels à travers le développement d'un service de diagnostic en ligne pour les composants logiciels embarqués et son intégration dans une plate-forme de gestion de réseaux de capteurs,
- les composants matériels et les interactions matériel-logiciel par la modélisation et la proposition d'une première approche de test statistique adaptée aux systèmes RFID et d'un middleware intégrant une approche de diagnostic probabiliste.

Les principales perspectives des travaux en cours sont décrites à la fin de chaque chapitre et une conclusion générale termine ce mémoire.

Chapitre 1 TEST ET DIAGNOSTIC EN LIGNE DE COMPOSANTS LOGICIELS EMBARQUES

I. Positionnement de l'étude

Le diagnostic de faute consiste à déterminer la nature d'une faute présente dans le système, sa localisation, sa date d'occurrence, *etc*. dans le but de permettre sa correction et la réparation du système. Les applications du processus de diagnostic regroupent des activités *hors ligne* telles que la maintenance et des activités *en ligne* comme la supervision d'un système et la tolérance aux fautes.

Les techniques et outils de diagnostic sont divers et variés et relèvent notamment de deux disciplines scientifiques : l'informatique et l'automatique avec parfois des interférences et des interactions entre les méthodes de l'une ou l'autre de ces disciplines. Ainsi, dans la typologie des approches de diagnostic relevant du contrôle-commande (automatique) proposée par Iserman [1], deux grandes familles de méthodes sont répertoriées : les méthodes de *classification* et celles fondées sur l'*inférence*. Cette typologie montre notamment comment des outils basés sur un formalisme mathématique, tels que l'intelligence artificielle, la logique floue ou encore les réseaux de neurones, se retrouvent dans l'une ou l'autre de ces deux familles de méthodes de diagnostic.

Dans nos travaux, nous nous intéressons essentiellement à des approches de *surveillance* ou *monitoring*, fondées sur l'observation continue ou périodique de paramètres prédéterminés du système. Le principe en est simple puisque toute valeur s'écartant du champ de validité d'un paramètre surveillé entraîne des actions complémentaires pour en connaître les raisons et déterminer les fautes éventuelles qui l'ont provoquée. A cette approche très répandue, se rajoute celle des *tests en ligne* qui a été largement étudiée dans le cas des composants matériels dans les systèmes multiprocesseurs et les systèmes répartis [2] [3] [4] et présentée comme une alternative moins coûteuse à la tolérance aux fautes basée sur les techniques de redondance. L'une des caractéristiques de cette approche consiste à utiliser des tests « légers », puisque les éléments d'un système en exploitation ont *a priori* été largement testés et validés avant leur utilisation en mode opérationnel. Ces tests en ligne mettent alors l'accent davantage sur des problèmes précis pouvant survenir pendant l'exécution et peuvent être considérés comme une approche complémentaire à celle de la supervision. La mise en œuvre de ces tests nécessite le plus souvent la soumission des cas de tests à l'élément sous test et la

récupération du résultat de test qui permet de déterminer l'état, correct ou défaillant, du composant sous test. La particularité des travaux cités ci-dessus consiste à introduire la notion de *test inter-composant* dans laquelle le composant testeur fait partie du système sous test. En effet, lors de l'utilisation du système, il n'est pas toujours aisé d'y introduire un équipement de test.

Dans nos travaux, nous nous sommes particulièrement intéressés à la sûreté de fonctionnement de systèmes hétérogènes incluant du matériel et du logiciel. Or, l'orientation du développement logiciel vers une approche par composant [5] en vue d'en simplifier la conception et l'évolution, a rapproché le développement logiciel de la conception de systèmes matériels par assemblage de composants préexistants. Cependant, la composition de composants logiciels ne permet pas de garantir que les attributs de qualité observés individuellement dans chaque composant sont conservés dans l'assemblage [6]. Par conséquent, il est crucial de doter les applications construites à base de composants logiciels de mécanismes leur permettant d'assurer un niveau de correction. L'analogie avec les systèmes matériels pousse naturellement à explorer les solutions développées et éprouvées pour les systèmes matériels et à analyser leur adéquation et leur adaptation aux applications logicielles à base de composants. Ainsi, parmi les approches proposées pour la sûreté de fonctionnement des applications logicielles à base de composants, nombreuses sont celles basées sur la réplication des composants et le masquage des fautes [7] [8] [9] [10] [11] [12] [13]. Les services fournis sont alors garantis corrects bien que certaines fautes puissent corrompre le fonctionnement de quelques composants applicatifs. De telles solutions peuvent néanmoins être coûteuses à cause de la réplication des composants. En effet, la réplication logicielle nécessite souvent des implémentations différentes d'un même composant, ce qui induit un coût assez important

Une alternative intéressante aux méthodes de réplication est l'utilisation des techniques de *Recovery Oriented Computing* (ROC) dont le but est de restaurer une exécution correcte du logiciel après l'occurrence de pannes, notamment en réparant les composants défaillants [14]. Pour cela, on utilise souvent un micro-redémarrage du composant tout en garantissant la cohérence avec le reste de l'application [14]. Mais il est nécessaire alors de connaître les composants défaillants pendant l'exécution. C'est à ce niveau qu'intervient l'utilisation des techniques de diagnostic citées ci-dessus afin de détecter et localiser les composants défaillants.

Comme nous le verrons dans la suite de ce chapitre, à l'instar des composants matériels, les composants logiciels peuvent également être dotés de capacités de test, s'autotester ou encore tester d'autres composants.

Ce chapitre comprend les parties suivantes :

- Une présentation de la notion de composant logiciel, avec une mise en évidence des éléments requis pour le test inter-composant en ligne ;
- Un bref aperçu du diagnostic système à base de tests inter-composants tel qu'il est mis en œuvre dans le matériel et une analyse de son adaptation aux composants logiciels ;

- Une étude du test de composants logiciels, plus spécifiquement lors de l'intégration ou pendant l'exécution, avec la motivation des choix effectués dans nos travaux ;
- La description des principales contributions de nos travaux dans le domaine du test et du diagnostic en ligne de composants logiciels ;
- Les principales perspectives de nos travaux, notamment dans une optique d'unification du test de composants logiciels et matériels.

II. Composants logiciels

II.1. Définitions et caractéristiques

Les composants logiciels sont des composants non physiques, d'où la difficulté rencontrée pour définir la notion de composant logiciel de manière universelle [15]. A cela s'ajoute une offre abondante de modèles et de plates-formes de composants logiciels, qui rend leur classification relativement peu aisée. Cette difficulté est contournée dans nos travaux par le recours à un ensemble minimal de caractéristiques communes à la plupart des modèles de composants courants, comme décrit dans le paragraphe II.2. du Chapitre I.

Deux définitions sont communément admises lorsqu'il s'agit de présenter la notion de composant logiciel. La première est celle de Szyperski [16] qui la définit comme suit : « Un composant logiciel est une unité de composition avec des interfaces contractuellement spécifiées et des dépendances explicites au contexte. Un composant logiciel peut être déployé indépendamment et est sujet à une tierce composition ». La seconde définition est celle de Heineman et Councill [5] : « Un composant logiciel est un élément logiciel conforme à un modèle de composant et qui peut être déployé et composé de manière indépendante sans modification, selon un standard de composition et d'interaction ». Ces deux définitions reprennent les principales caractéristiques d'un composant logiciel, à savoir :

1. Un composant est considéré comme étant une entité logicielle possédant des interfaces bien définies et ayant des dépendances exprimées clairement (de manière contractuelle).
2. Il peut être composé et déployé indépendamment dans un système.
3. Un composant logiciel est conforme à un modèle de composant qui définit ses caractéristiques, la composition de plusieurs composants et le support d'exécution correspondant.

Un modèle de composants logiciels permet donc de définir : i) les règles pour la construction de composants individuels, ii) les règles pour l'assemblage de ces composants dans un système.

Une classification relativement exhaustive des modèles de composants logiciels est présentée dans [15]. Elle se fonde sur trois critères :

1. Le *cycle de vie*, qui englobe les aspects de modélisation, d'implémentation, d'empaquetage et de déploiement des composants. Ces étapes permettent principalement de décrire les aspects architecturaux des composants et leurs interactions, de générer le code correspondant, de produire les paquetages nécessaires

à leur diffusion et enfin de déterminer les actions nécessaires à leur intégration dans un environnement cible.

2. La *construction* (ou *composition*) : ce volet s'intéresse aux aspects liés aux interfaces des composants, aux mécanismes d'établissement des connexions entre les composants et à la mise en œuvre pratique des interactions ou communications entre composants.
3. Les *propriétés non fonctionnelles* : cet aspect concerne la spécification et la gestion des propriétés non fonctionnelles rattachées aux composants.

Quel que soit le modèle de composant retenu, rendre un composant testable implique d'intervenir sur chacun de ces critères dans le but d'automatiser l'intégration de la capacité d'être testé et de tester et éventuellement de la rendre transparente vis-à-vis de l'utilisateur. Or, il existe un grand nombre de modèles de composants logiciels. La synthèse présentée dans [15] les classe en trois principales catégories :

- les *modèles génériques* : ces modèles fournissent les mécanismes de base pour la spécification et la composition des composants sans préciser l'architecture requise pour leur utilisation. Les modèles de composants CCM [17], EJB [18], Fractal [19] et Koala [20] font partie de cette classe de modèles.

- les *modèles spécifiques* qui prennent en considération les contraintes d'un domaine métier particulier et y répondent efficacement mais sont par conséquent difficilement adaptables à d'autres domaines. Les modèles AUTOSAR [21]**,** OPC [22] et OSGi [23], dédiés respectivement aux applications automobiles, aux applications de contrôle automatique et aux systèmes embarqués, sont dans cette catégorie de composants.

- les *modèles de composants générateurs* dont le but est d'instancier des modèles de composants particuliers. Les modèles de composants Fractal [19] et ROBOCOP [24] sont des exemples de tels modèles. Un modèle générateur peut également se retrouver dans l'une des deux catégories précédentes.

Dans nos travaux, la capacité des composants logiciels à interagir avec d'autres composants dans le cadre d'une relation de test inter-composant entraîne une vision légèrement différente, des caractéristiques des composants. Dans nos travaux, ceux retenus sont [25] :

1. La *définition* du composant et de ses interfaces : concerne le développement du composant et ses frontières avec le monde extérieur.
2. La *composition* des composants : il s'agit de l'interaction entre les composants ; et la manière dont les composants sont assemblés pour former des applications.
4. Le *framework* du composant : c'est l'environnement d'exécution du composant qui supporte l'exécution des composants en gérant les interactions entre composants et l'invocation des différents services fournis par les composants.

Ainsi, connaître la définition d'un composant et de ces interfaces nous permet de savoir comment mettre en place le test d'un composant. L'analyse de l'interaction entre les composants (ou composition des composants) est nécessaire pour développer la notion de test inter-composant en ligne. Enfin, l'environnement d'exécution du composant est important

pour la mise en œuvre des tests inter-composants et du diagnostic en ligne des éléments défaillants. Nous pouvons remarquer que les éléments introduits par [15], notamment le cycle de vie et les propriétés fonctionnelles, sont également considérés, de manière indirecte, dans les critères de classification retenus dans nos travaux.

Une étude comparative de quelques modèles de composants selon les critères ci-dessus est présentée dans la thèse de T.-Q. Bui [25]. Cette étude nous a principalement permis de déduire les caractéristiques minimales requises d'un modèle de composants logiciels en vue du développement d'une approche de test en ligne. Ainsi, nous avons pu nous affranchir de tout modèle de composant spécifique.

II.2. Modèle considéré de composants logiciels

L'objectif de nos travaux sur la sûreté de fonctionnement des applications à base de composants logiciels est de proposer une approche générique pouvant s'adapter à tout modèle de composant logiciel. Nous nous sommes donc appuyés sur un modèle de composant général, dans lequel la seule contrainte concerne l'existence d'interfaces *requises* et d'interfaces *fournies* pour permettre de définir la composition des composants, notamment pour ce qui concerne les relations de test inter-composants. La figure 1.1. ci-dessous représente un tel composant.

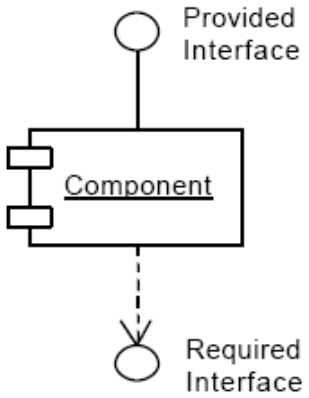

Figure 1.1. Un composant logiciel avec une interface fournie et une interface requise

Le paragraphe suivant introduit les concepts de base du diagnostic système à base de tests inter-composants.

III. L'approche de diagnostic système à partir de test inter-composants

III.1. Définitions

Le diagnostic en ligne à partir de tests inter-composants, initialement proposé pour les *systèmes matériels* par Preparata *et al.* [26], permet de déterminer l'état d'un système vis-à-vis des fautes pouvant toucher ses composants. L'idée fondamentale est d'utiliser des tests inter-composants. Un composant est alors utilisé pour tester et déterminer l'état (fautif ou correct)

d'un autre composant. Le processus de diagnostic consiste donc à analyser les résultats des tests pour en déduire l'état de tous les composants du système. Or, il est possible qu'un composant testeur soit lui-même défaillant et transmette par conséquent des résultats de test non fiables. L'enjeu principal du processus de diagnostic consiste alors à analyser l'ensemble des résultats de tests produits tout en éliminant ceux provenant de composants testeurs défaillants. Le fait que ceux-ci ne sont pas connus *a priori* représente la difficulté majeure de ce processus.

Ce problème a donné lieu à de nombreux travaux, essentiellement théoriques, dont quelques états de l'art ont été publiés [2] [3] [4] et dont les principaux travaux récents sont [27] [28] [29] [30] [31] [32] [33] [34]. A notre connaissance, cette problématique a été peu étudiée dans le cas des composants logiciels, contrairement à celle du test des composants logiciels, dont les travaux se sont largement inspirés de l'existant dans les composants matériels (*cf.* Chapitre 1, section IV). Nous nous limitons donc à présenter ci-dessous les principales caractéristiques des algorithmes de diagnostic proposés pour les systèmes matériels dans le but d'en extraire les caractéristiques pouvant être adaptées aux composants logiciels.

Le processus de diagnostic peut être *centralisé* ou *distribué*. Dans le diagnostic centralisé, un composant central supposé fiable contrôle l'exécution des tests inter-composants et calcule l'état du système à partir des résultats des tests. Le diagnostic distribué permet à tout composant du système de déterminer l'état du système complet à partir des informations de tests produites localement et de celles transmises par d'autres composants.

Selon la manière dont les tests sont affectés[1], les algorithmes de diagnostic peuvent être *statiques* ou *adaptatifs*. Dans les algorithmes statiques, les relations de test sont établies avant le processus de diagnostic. Elles sont dynamiques dans la stratégie adaptative et dépendent des résultats de tests précédents. L'avantage principal des algorithmes adaptatifs est l'utilisation d'un nombre de tests réduit qui entraîne logiquement une réduction du nombre de messages échangés et du temps d'exécution du diagnostic.

Nous présentons ci-dessous le principe de fonctionnement du diagnostic centralisé et du diagnostic distribué sachant que de nombreux algorithmes représentatifs de ces deux types de diagnostic ont été proposés. Il est important de noter que les principes de base présentés ci-dessous se fondent sur les deux hypothèses suivantes :

- Les fautes qui surviennent au niveau des composants sont permanentes.
- Un test exécuté par un composant fiable produit toujours un résultat correct et indique avec certitude l'état du composant testé, alors qu'un test exécuté par un composant défaillant n'est pas fiable et peut fournir des informations erronées sur l'état du composant testé.

III.2. Principe du diagnostic centralisé

Le principe du diagnostic centralisé est simple. En plus de supposer l'existence d'un élément *central fiable*, ce diagnostic fait l'hypothèse de la présence simultanée d'un *maximum*

[1] Les relations de test entre les composants forment un graphe orienté, appelé *graphe de test*.

de t composants défaillants durant une session de diagnostic. Afin d'assurer la détection et la localisation des t composants défaillants, il est nécessaire et suffisant que chaque composant soit testé par au moins t autres composants différents, sachant que $N \geq 2t+1$, où N est le nombre total de composants dans le système.

Le composant central, supposé fiable, assure les actions suivantes :

a. détermination du graphe de tests (relations de tests inter-composants)
b. transmission des assignations de tests aux différents éléments du système,
c. collecte des résultats de tests,
d. diagnostic centralisé.

La satisfaction de la condition ci-dessus sur le nombre d'éléments fautifs garantit que le diagnostic établi est correct. En effet, un consensus est possible et est réalisé car les composants corrects sont majoritaires.

III.3. Principe du diagnostic distribué

Le diagnostic distribué permet à tout composant de déterminer l'état du système à partir d'informations de test produites localement et de celles transmises par d'autres composants. Deux aspects doivent être résolus dans ce cas :

a) *la distribution des tests* : dans le diagnostic distribué, le contrôle des tests se faisant par les composants eux-mêmes, il est important de disposer de stratégies de test efficaces qui permettent à tout composant de déterminer l'ensemble des composants qu'il doit tester,

b) *la transmission des informations de diagnostic* : afin de permettre à tout composant correct du système de déterminer l'état du système, il est nécessaire que les composants s'échangent entre eux les informations de diagnostic engendrées localement. Par ailleurs, la charge induite par les messages de diagnostic ne doit pas compromettre les performances du système.

L'idée principale du diagnostic distribué est qu'un composant n'accepte que les informations en provenance des composants qu'il teste et qu'il trouve corrects. Globalement, un composant opère de manière périodique selon le schéma suivant (les N composants du système sont numérotés C_0, C_1, C_2, …, C_{N-1}) :

- le composant C_i teste le composant C_j comme étant correct,
- le composant C_i reçoit des résultats de tests transmis par le composant C_j,
- le composant C_i teste une nouvelle fois le composant C_j,
- le composant C_i suppose que les informations de tests transmises par C_j sont valides si C_j est toujours correct.

Selon ce principe, le diagnostic distribué peut diagnostiquer jusqu'à N-1 composants fautifs simultanés, N étant le nombre de composants dans le système. Pour cela, l'hypothèse d'une interconnexion complète des composants est faite et une stratégie de diagnostic adaptative est utilisée. Le diagnostic distribué nécessite globalement moins de tests inter-composants que le diagnostic centralisé

En bref, nous pouvons constater que 2 types d'hypothèses sont nécessaires au fonctionnement correct d'un algorithme de diagnostic :

- Une hypothèse sur la nature des fautes considérées.
- Une hypothèse sur la connectivité du système qui conditionne le nombre de composants pouvant être simultanément défaillants.

La nature du diagnostic (centralisé ou distribué) et la stratégie de test utilisée (statique ou adaptative) sont également des éléments importants.

Si la question de la connectivité se pose naturellement pour les composants matériels, elle est d'une moindre importance dans le cas des composants logiciels. En effet, l'interconnexion de ceux-ci étant définie au niveau applicatif, il est possible de la considérer comme état complète, amenant ainsi les capacités de diagnostic à leur maximum. Par ailleurs, dans nos travaux sur les composants logiciels, seules les fautes permanentes ont été considérées. Les fautes intermittentes sont considérées dans l'étude des systèmes RFID présentée dans le Chapitre 2 de ce mémoire.

IV. Test de composants logiciels

IV.1. Etapes et difficulté du test de composants

Le processus de test est un processus de vérification et de validation qui permet de détecter la présence de fautes dans le système sous test et de s'assurer que le comportement du système sous test est conforme à ses spécifications. Dans le cas du logiciel, cela revient à exécuter un ensemble de cas de tests. Selon les standards IEEE, un cas de test documente les entrées ou données du test, les résultats attendus et les conditions d'exécution du test. Ainsi, un programme sous test est exécuté avec des entrées valuées, et la conformité des sorties obtenues avec les résultats attendus est vérifiée [35]. Etant donné un modèle de fautes précis, la mise en œuvre du test nécessite les opérations suivantes :

1. La détermination des niveaux de test (test unitaire, test système, *etc.*)
2. La sélection et la génération des cas de test,
3. La vérification de l'exactitude des résultats observés (*problème de l'oracle*).

Classiquement, une connaissance précise des attentes de l'utilisateur vis-à-vis du logiciel et de l'environnement futur d'exploitation du logiciel permet de mener à bien les différentes étapes du test. Or, le développement de composants logiciels se fait sans une connaissance détaillée de l'environnement final d'utilisation du composant, et ce à plus forte raison que le même composant est appelé à être intégré dans diverses applications. Par conséquent, certains chercheurs se sont intéressés au problème crucial du manque d'information sur l'environnement final des composants et de son incidence sur le test de ces composants. En particulier, les travaux de Beydeda et Gruhn [36] ont largement considéré le manque d'informations lors du test d'un composant et les implications que cela peut avoir sur le processus de test. Ces auteurs considèrent que le manque d'informations rend nécessaire des tests par l'utilisateur du composant, pouvant contredire ainsi le fait qu'un composant validé

par son fournisseur n'a pas besoin d'être testé de nouveau par l'utilisateur. Ainsi, même s'il est entendu que le processus de test soit partie intégrante du développement d'un système (logiciel ou matériel ou mixte), il est rapidement apparu que dans le cas du développement logiciel à base de composants, une phase de test lors du déploiement de l'application est nécessaire. Une raison majeure à cela est mentionnée dans les travaux de Beydeda et Gruhn [36] qui indiquent que le développeur d'un composant est souvent amené à établir des hypothèses sur l'environnement d'utilisation futur du composant. Ceci rend les tests menés par le développeur dépendants de ces hypothèses et a pour conséquence notamment la nécessité de nouveaux tests par l'utilisateur pour s'assurer de la conformité des composants vis-à-vis de l'environnement d'exécution.

Classiquement, les phases de test lors d'un développement à base de composants logiciels sont les suivantes :

- *Test unitaire* : ce test peut se faire en boîte blanche et permet de s'assurer que le composant ne comporte pas d'erreur. Il peut également se dérouler en boîte noire pour s'assurer que le composant réponde bien aux spécifications qui lui sont attachées. Le test unitaire ne peut toutefois pas vérifier que le composant réponde fonctionnellement aux demandes concernant le système dans lequel il sera intégré.
- *Test d'intégration/de déploiement* : ce type de test concerne les interactions des composants entre eux comme envisagé dans l'environnement final avant le déploiement. Le test de déploiement se déroule généralement en deux phases : une phase durant laquelle le composant est testé dans un environnement simulant l'environnement final (en utilisant des *stubs* pour simuler les composants manquants) et une phase dans laquelle quelques composants sélectionnés sont assemblés et testés entre eux. A ce niveau, les erreurs détectées sont souvent dues à une non-conformité des composants par rapport aux spécifications du système.
- *Test système* : ce test se déroule lorsque tous les composants du système sont intégrés. Il est à la charge de l'utilisateur. L'objectif est ici de tester les fonctions du système en tant que boîte noire, en tenant compte des contraintes de charge et de performances.
- *Test de régression de composant* : permet le test de composants déjà testés, souvent dans un nouveau contexte ou suite à des changements intervenus dans les composants.

Cependant, les divers cas d'utilisation que rencontre le composant durant son exploitation ne peuvent raisonnablement être couverts par le test de déploiement, d'autant que les composants sont appelés à évoluer dans des environnements dynamiques, mais également à évoluer eux-mêmes de manière dynamique. Par ailleurs, durant leur cycle de vie, les composants peuvent subir des fautes suite à des interactions avec d'autres composants ou en raison de défauts dans la plate-forme d'exécution sous-jacente. Par souci de continuité de service et de tolérance aux fautes, des *tests en ligne* lors de l'utilisation des composants nous semblent être une solution souple et efficace.

A notre connaissance, peu de travaux ont étudié le test en ligne des composants logiciels, alors que le test d'intégration et de déploiement a suscité un intérêt certain de la communauté scientifique depuis une quinzaine d'années. Les paragraphes ci-dessous résument les

principales approches existantes pour le test de déploiement et le test en ligne de composants logiciels.

IV.2. Test d'intégration ou de déploiement

IV.2.1. Principales approches du test d'intégration

Certains auteurs ont établi un état de l'art des approches de test existantes pour les composants logiciels. La synthèse de Beydeda et Gruhn [36] s'est principalement focalisée sur les approches qui tiennent compte du manque d'informations liées aux composants et qui peuvent être utilisées par les utilisateurs des composants. Ces approches sont classées en deux catégories : celles dont le but est *d'éviter le manque d'information*, traitant ainsi la cause du problème et celles qui *traitent les conséquences du manque d'informations* sur le test par les utilisateurs.

- *Approches traitant la cause du manque d'informations* : l'idée ici est de favoriser la transmission d'informations entre le fournisseur et le client. Les principales solutions se fondent sur l'utilisation des *métadonnées* qui font partie du paquetage des composants, pour assurer la communication dans le sens fournisseur-client. Elles peuvent également utiliser la *réflexivité* qui dote les composants de capacités d'auto-description utiles dans ce contexte. Les informations ainsi fournies peuvent être exploitées par l'utilisateur des composants pour les tester lors du déploiement.

- *Approches traitant les effets du manque d'informations* : ces approches constituent la classe du « *Built-in* » test et ont pour principe commun de doter les composants de capacités de test. Le principal inconvénient est ici la surcharge des composants avec des informations qui ne servent que pendant la phase de test. Certains travaux définissent alors un *mode test* qui permet l'inclusion des mécanismes de test uniquement quand cela est nécessaire [37] [38] [39]. D'autres travaux, définissent des types précis de composants, utilisables pour le test uniquement. Par exemple, dans l'approche Component+ [40] [41] [42], trois types principaux de composants sont définis : composants *BIT* devant être testés, composants *testeurs* qui comportent les cas de tests pour tester les composants BIT et composants *gestionnaires* (ou *handlers*) qui sont utilisés pour le recouvrement d'erreurs. Dans l'approche « *testable beans* » [43] qui concerne les composants EJB, une interface de test comportant trois méthodes est ajoutée à chaque composant : une méthode sert à initialiser le test, une autre pour l'exécuter et la dernière pour l'évaluer.

Il découle de cette première classification que les informations utiles au test sont une question centrale. En effet, les composants sont destinés à être fournis avec une interface précisant leurs services. La problématique du test lors de leur déploiement ou pendant leur exécution est ainsi rarement résolue pendant leur développement.

C'est ainsi que tout en considérant les informations utiles au test, une autre classification des approches de test proposée par [44] considère les informations que le composant lui-même comporte en vue de son test par l'utilisateur ou un tiers (autre que le développeur). Il ressort de cette seconde classification cinq catégories d'approches, dont certaines se recoupent avec celles proposées dans la classification précédente :

1. Les approches de test intégré (Built-in testing ou BIT approaches)

Tout comme dans le cas d'un composant matériel, le test intégré d'un composant logiciel revient à augmenter ce composant du code et des données nécessaires au test (ce que l'on désigne par la *logique de test* dans le matériel). Ainsi, outre les éléments fonctionnels, un composant contient selon les approches de test intégré, des éléments qui permettent de le tester par l'utilisateur final. Les approches de test intégré ont été analysées dans [44] et sont décrites dans le paragraphe IV.2.2 du Chapitre 1 ci-dessous.

2. Les architectures testables (Testable architecture ou auto-test)

Dans ce cas, les tests sont fournis sous forme de spécifications et non pas d'éléments rajoutés au composant. Ceci entraîne un gain en mémoire. Les travaux décrits dans [40] [43] sont représentatifs de cette catégorie d'approches.

3. Les approches basées sur les métadonnées (Metadata-based approaches)

Le principe de ces approches est l'ajout d'informations permettant de faciliter le test du composant par l'utilisateur. Ces informations prennent la forme de métadonnées, ce qui évite de divulguer le code du composant. Les formes les plus utilisées pour fournir ces informations sont les *machines à états finis* pour représenter un modèle du composant, ou des *pre* et *post conditions* sur les services fournis par le composant. Dans ce cas, les outils nécessaires pour gérer ces métadonnées en vue du test doivent être disponibles pour permettre à l'utilisateur de générer des cas de tests. Des exemples de travaux utilisant les métadonnées sont présentés dans [45] [46].

4. La certification (Certification strategy)

L'objectif de ces approches est d'augmenter la confiance de l'utilisateur dans le composant en faisant passer celui-ci par un processus de *certification reconnu*. Un tel processus peut être mené par un tiers, par le développeur du composant ou par l'utilisateur lui-même. Ces approches engendrent un certain coût, en particulier si elles sont menées par un tiers et sont souvent réservées aux composants critiques [47].

5. Le test basé sur les spécifications de l'utilisateur (User's specification-based testing)

Dans ces approches, l'utilisateur s'affranchit complètement du développeur du composant et se base sur les spécifications du composant pour mener des tests en boîte noire [48].

Il apparaît clairement à partir des éléments ci-dessus que la question des informations utiles au test des composants logiciels est cruciale et qu'elle a orienté les recherches dans ce domaine. Cependant, la mise en œuvre des solutions adoptées est tout aussi importante, surtout lorsqu'un test en ligne est envisagé. A ce stade de notre analyse, les questions qui se posent concernent les aspects suivants :

- Quelles informations associer au composant pour en assurer le test dans un contexte en ligne et sous quelle forme : métadonnées, enveloppes de composants, spécifications de tests, *etc*. ?
- Mise à disposition de ces informations : ces informations font-elles partie du composant ou sont-elles activées uniquement pour le test ?

- Un framework pour la gestion des tests est-il nécessaire et si oui, sous quelle forme doit-il se présenter ?

Pour assurer le test en ligne des composants logiciels, ceux-ci doivent être rendus *testables* ou *auto-testables*, tout comme le sont les composants matériels. Par ailleurs, la prise en charge des tests par l'utilisateur assure une meilleure efficacité et adéquation de ceux-ci au contexte réel d'utilisation des composants. Même s'il est important de préserver l'indépendance de l'utilisateur vis-à-vis du développeur en ce qui concerne le test des composants, le développeur doit garantir l'inclusion d'informations minimales dans le composant pour en permettre le test par l'utilisateur. A cet égard, les spécifications du composant nous semblent être une demande réaliste.

Les approches basées sur les architectures testables et les métadonnées imposent une forte dépendance de l'utilisateur vis-à-vis du développeur alors que les approches de certification présentent l'inconvénient prohibitif d'un coût trop élevé. Par conséquent, l'orientation de nos travaux s'est principalement focalisée sur les approches de *test intégré* ou *Built-In-Test* qui offrent un large choix de techniques pour rendre les composants testables, comme cela est décrit dans le paragraphe IV.2.2. du Chapitre 1.

En outre, les approches de test intégré (BIT) sont celles qui se sont largement inspirées du matériel. Nos objectifs de recherche à plus long terme comprenant une unification du traitement des composants matériels et logiciels, il nous a semblé que les approches de test intégré sont les plus susceptibles de nous permettre la concrétisation d'un tel objectif. Ainsi, nous avons opté pour une étude approfondie des approches de test intégré des composants logiciels qui s'inspirent largement de l'existant au niveau du matériel, sachant que l'on retrouve souvent dans ces approches l'utilisation de métadonnées et d'architectures testables.

IV.2.2. Les différentes approches de Built-In-Test dans les composants logiciels

Dans ces approches, les éléments additionnels utiles au test peuvent prendre diverses formes, parmi lesquelles :

1. Une *enveloppe de composant ou BIT wrapper* dans les travaux de Edwards [49]. Cette enveloppe se présente sous la forme de spécifications incluant des *pre* et *post* conditions pour chaque interface du composant. Elle agit comme un filtre pour détecter toute violation de l'interface du composant. Un framework pour la construction de ces enveloppes de composants en vue du test est proposé et permet de générer à partir des spécifications du composant et de ses interfaces, les cas, les pilotes et les oracles nécessaires au test. Cependant, cette approche dépend du développeur pour fournir l'enveloppe de test intégré [44].

2. Des *fonctions de test additionnelles* dans les travaux de Wang *et al*. [38] [39]. Un composant peut fonctionner selon cette approche en deux modes, *mode normal* et *mode de maintenance*. En mode normal, les possibilités de test intégré sont transparentes à l'utilisateur du composant et le composant est identique à un composant sans test intégré. Cependant, en mode d'entretien ou de maintenance, l'utilisateur du composant peut tester le composant avec

l'aide du test intégré. L'utilisateur du composant peut alors appeler des méthodes du composant qui exécutent le test, évaluent de façon autonome ses résultats, et fournissent un rapport de test. Les cas de test et leurs implémentations sont fournis par le développeur du composant. Ils peuvent être contenus dans le composant ou générés à la demande. Avec ces cas de test, le composant peut se tester lui-même ou l'utilisateur peut exécuter les cas de test via l'interface de test. L'avantage d'une telle approche est de permettre l'accès aux variables privées du composant et de ne pas nécessiter d'outil ou de framework spécifique pour l'exécution [44]. L'inconvénient principal de cette approche concerne le stockage des cas de test ou d'une description de leur génération dans le composant. Ceci peut rendre le composant coûteux, notamment par ses demandes accrues de ressources. De même, il y a une forte dépendance avec le développeur. Par ailleurs, un tel test intégré a été envisagé uniquement lors du déploiement des composants. Le coût induit n'est donc pas amorti par une utilisation plus fréquente comme envisagé dans le test en ligne.

3. Des *interfaces additionnelles* dans le cas des travaux de Martins *et al*. [37] qui s'inspirent des approches de *conception en vue du test* dans le matériel dont le but est d'augmenter l'*observabilité* et la *contrôlabilité* d'un composant matériel par l'ajout de broches (pins). L'approche proposée nécessite une étroite collaboration entre le fournisseur et le client. Ainsi, il est attendu du fournisseur de construire le modèle de test, de développer la spécification de test, notée *t-spec*, à partir de ce modèle et de l'insérer dans le code source du composant, d'instrumentaliser le code source en intégrant des mécanismes de built-in test. Le client doit quant à lui générer les cas de test à partir de la t-spec fournie, compiler le composant en mode test, exécuter les tests et analyser les résultats obtenus. Cette approche nous semble particulièrement adaptable au test en ligne des composants logiciels, même si dans le cadre de nos travaux les actions assurées par le développeur doivent être à la charge de l'utilisateur.

4. Des *cas de test et du code de test intégré* dans l'approche proposée par Beydeda [50] [51] sous l'appellation STECC (Self-Testing COTS Component). Ces éléments de test sont issus du test unitaire du composant et intégrés en vue de leur réutilisation pour le test d'intégration. Ainsi, le composant peut se tester lui-même sans besoin d'exporter son code source. Cette approche utilise les tests basés sur le programme, c'est-à-dire des tests structurels. Ainsi, chaque composant encapsule un graphe de flot de contrôle qui modélise les informations du code source. Les cas de test sont générés et exécutés par le framework STECC. Le framework STECC implémente plusieurs algorithmes pour déterminer les chemins à traverser selon les critères spécifiques et pour générer les cas de test correspondants. Tout comme les approches précédentes, il y a une forte dépendance avec le développeur.

5. Une *librairie Java* dans les travaux de Groß *et al*. [52] menés dans le cadre du projet Component+ qui utilisent les propriétés de réflexion et d'introspection de ce langage pour permettre l'accès aux caractéristiques des composants sous test. Le test développé par ces auteurs est un test basé sur la notion de *contrat* permettant de vérifier qu'un composant remplit ses contrats vis-à-vis des composants qui l'utilisent.

6. Un *framework de test* dans le cas du projet Component+ également [40] [41] [42]. Ce framework comporte trois types de composants : les composants BIT, les testeurs et les

gestionnaires. Les composants BIT sont les composants dotés de test intégré. Ces composants implémentent certaines interfaces obligatoires et rejoignent en cela les travaux de Martins *et al*. sur les composants auto-testables [37]. Les testeurs sont les composants qui accèdent aux possibilités de test intégré des composants BIT par les interfaces correspondantes et qui contiennent les cas de test. Enfin, les gestionnaires sont des composants qui ne contribuent pas au test, mais sont nécessaires, par exemple, pour assurer les mécanismes de recouvrement en cas de défaillances. Ce framework a principalement été utilisé dans le cadre d'un test de Qualité de Service (QoS testing) dont le but est de s'assurer que le composant se comporte correctement à l'exécution et d'identifier les points où d'éventuels défauts peuvent survenir. Le test de QoS est assuré ici sur la base de contrats.

Du fait de leur variété et de leur diversité, il est relativement difficile de comparer efficacement les travaux sur le test intégré ou BIT des composants logiciels. En effet, chaque approche peut convenir dans un contexte donné en fonction des informations disponibles à l'intégrateur ou de la criticité de l'application cible. Toutefois, dans la perspective d'une adaptation à un test en ligne, deux critères importants concernent la *charge de travail* qui incombe à l'utilisateur final et son *autonomie* vis-à-vis du développeur. Ces deux critères sont contradictoires dans une large mesure. En effet, une forte dépendance de l'utilisateur vis-à-vis du développeur peut réduire la charge de travail qui incombe à l'utilisateur. Mais elle a l'inconvénient de ne pas garantir une adéquation totale avec les contraintes de l'environnement qui ne sont connues que tard dans le cycle de vie du composant. Une option intéressante serait de minimiser la dépendance de l'utilisateur mais en lui fournissant les outils nécessaires pour réduire la charge de travail liée à la mise en place des tests. Cela revient à assurer l'*automatisation* de la mise en place des tests (génération des cas de tests, exécution des tests, analyse des résultats des tests, *etc*.). Dans nos travaux, l'absence d'hypothèses précises sur la nature du modèle de composant rajoute une contrainte supplémentaire en ne rendant éligibles que les solutions présentant une certaine indépendance vis-à-vis du modèle de composant. Ces contraintes éliminent ainsi toutes les solutions basées sur les métadonnées, les enveloppes de tests, l'inclusion par le développeur d'informations sur le test ou encore les propriétés de réflexivité et d'introspection. Ainsi, seules les approches basées sur des interfaces et des fonctions de test, ou l'utilisation de frameworks dédiés au test semblent facilement adaptables au test en ligne. Rappelons que l'unique hypothèse formulée dans nos travaux concerne la transmission par le développeur des spécifications du composant. Ces spécifications sont alors utilisées pour la génération des cas de test et l'exécution des tests inter-composants. Il est par conséquent important de fournir un framework permettant de réaliser ces différentes étapes.

Le paragraphe ci-dessous clôt notre revue des travaux existants dans le test de composants logiciels par quelques études publiées sur le test en ligne du logiciel. Comme on le verra, ces études ne concernent pas la sûreté de fonctionnement, mais principalement des aspects de liaison dynamique des composants.

IV.3. Test en ligne de composants logiciels

A notre connaissance, les principales utilisations du test en ligne du logiciel sont liées à l'intégration et la liaison dynamique de nouveaux composants ou de nouvelles fonctionnalités tout en conservant en activité l'application concernée. Ainsi, le test en ligne dans le cadre de l'expansion d'une application existante par l'intégration en ligne de nouvelles fonctionnalités logicielles ou l'évolution de fonctionnalités existantes est relaté dans [53]. Ce travail ne concerne pas spécifiquement les applications à composants, mais plutôt les applications distribuées comportant des contraintes temps réel. Les tests en ligne mis en place ont alors pour objectif de vérifier le fonctionnement correct des nouvelles fonctionnalités tant du point de vue de l'exécution que du respect des contraintes temporelles. Les auteurs mentionnent, sans toutefois le mettre en pratique, l'utilité de tels tests pour la tolérance aux fautes. D'autres travaux ont étudié l'utilisation de tests en ligne lors de la liaison dynamique de nouveaux composants, selon deux principales approches : celle fondée sur la détection des incompatibilités au moment où elles se produisent comme c'est le cas dans le projet européen PLASTIC [54] qui s'intéresse au développement de services logiciels pervasifs robustes et l'approche prônant la détection *a priori* des incompatibilités [55]. L'objectif est alors ici limité à la détection des incompatibilités (d'interfaces par exemple) qui pourraient être créées par l'introduction dynamique de nouveaux composants, sans toutefois étendre cela à une prise en charge plus large de la tolérance aux fautes.

Les auteurs dans [56] décrivent l'architecture d'une plate-forme de test en ligne générale en spécifiant les différents éléments qu'une telle plateforme devrait proposer et en prenant pour cas d'étude un système à base de composants. Les principaux éléments cités sont :

1. ***Des sondes (probes)*** : ces éléments instrumentent le système sous test aux points d'observation, pour extraire les informations utiles au test. Ce sont en général des bouts de logiciel, dépendants ou indépendants des composants sous test, insérés de façon manuelle ou de manière automatique.

2. ***Des injecteurs (injectors)*** : permettent l'injection de fautes dans les composants sous test.

3. ***Des cas de test en ligne (on line test cases)*** : vérifient le comportement attendu des composants sous test et signalent les cas de non-conformité.

4. ***Un contrôleur de configurations (configuration controller)*** : gère la coordination du système de test en ligne (création et suppression de sondes, de cas de tests, *etc*.)

5. ***Un bus de communication (communication bus)*** : réalise les fonctions de communication en transmettant à chaque composant du système de test en ligne les événements qui l'intéressent (un fonctionnement de type abonnement est alors préconisé).

6. ***Un modèle de système (system model)*** : des informations sur le système sous test sont nécessaires pour le fonctionnement du système de test en ligne. Ces informations sont stockées dans une base de données. Tout changement du système sous test est répertorié dans cette base de données. Ce modèle doit aussi permettre de stocker des informations sur le comportement dynamique du système sous test, d'où l'importance de stocker les données

relatives aux événements. Ceci permet aussi de faire des analyses sur l'historique du système sous test.

7. ***Des jauges (gauges)*** : ce sont des composants logiciels qui captent les événements sur le bus et les enregistrent pour maintenir une vue à jour du système sous test.

Etant générale, la description ci-dessus peut s'appliquer aisément à un test en ligne de composants logiciels mais nécessite cependant de spécifier et définir plus précisément les différents éléments mis en œuvre. En particulier, il est nécessaire de savoir précisément comment seront réalisées : la génération des cas de test, l'exécution des tests et l'analyse des résultats.

Le test en ligne de composants logiciels est également étudié dans [57] et concerne plus spécifiquement les composants CORBA ou COM/DCOM et la technologie Java. Les auteurs spécifient la plateforme à mettre en place en utilisant le paradigme client-serveur pour l'exécution des tests. Comme vu précédemment pour le test de déploiement, il s'agit aussi d'une technique BIT, mais dans ce travail un composant client invoque le test d'un composant serveur. Ainsi, le modèle client-serveur sert de base à la relation de test entre deux composants donnés. Le composant serveur sous test comporte dans son interface les méthodes nécessaires à son test. Il n'y a donc pas d'interface de test spécifique. Les auteurs insistent principalement sur la mise en œuvre des éléments nécessaires au test, mais ne précisent pas comment gérer l'exécution des tests en ligne et leur interférence avec les fonctionnalités des composants.

Enfin, il est utile de noter que les travaux menés dans le cadre du projet Component+ précédemment cité pour le test d'intégration ont également concerné un volet lié au test en ligne, mais limité à un test de qualité de service. Les aspects tolérance aux fautes n'y sont pas traités.

IV.4. Conclusion

Les travaux sur le test des composants logiciels présentés ci-dessus ont pour principal objectif de tester un composant dans son nouvel environnement d'exécution lors du déploiement d'une application à base de composants. Dans ce cas, les composants sont comme des boîtes noires ce qui rend nécessaire le recours à des tests fonctionnels définis à partir des spécifications des composants, dont la disponibilité est souvent requise. Dans la plupart des approches de test intégré, les cas de test utilisés sont développés par le fournisseur du composant, ce qui peut limiter leur intérêt pour l'utilisateur. Par ailleurs, les éléments rajoutés en vue du test engendrent un coût à l'utilisation, en particulier au niveau de la mémoire pour ce qui est du test de déploiement. En outre, dans le cas de l'utilisation visée de ces approches, les éléments rajoutés en vue du test ne sont exploités que lors de l'intégration et sont donc inutiles pendant l'exécution des composants.

Le test de déploiement peut ne pas suffire à révéler tous les défauts résiduels dans les composants et certains de ces défauts peuvent se manifester lors de l'exécution lorsque les conditions de leur activation sont réunies. De plus, les composants logiciels sont appelés à s'exécuter dans des environnements dynamiques, qui évoluent au cours du temps. Comme

cela a été vu à travers les travaux de test en ligne du logiciel, la fonctionnalité de test en ligne est principalement utilisée lors de la mise à jour de composants ou de la liaison dynamique de nouveaux composants. Son utilisation pour assurer la tolérance aux fautes, bien qu'évoquée, n'a à notre connaissance pas été abordée.

Dans nos travaux, le test de composants doit servir le processus de diagnostic en ligne lors de l'exécution des applications. Notre intérêt se porte donc principalement sur une exécution en ligne du test des composants. En plus, les applications à base de composants ne disposent pas toujours du code des composants. De ce point de vue, les divers mécanismes introduits dans le cadre du test intégré pour le déploiement nous semblent réutilisables et adaptables à des degrés divers et variés. Notre choix s'est porté sur les approches BIT pour les raisons précédemment évoquées. Les principales contributions de nos travaux consistent donc à rendre possible la mise en place du test en ligne en partant des spécifications du composant. Cette étape doit être largement automatisée pour minimiser la charge de travail de l'utilisateur. En outre, il est important de tenir compte des contraintes de concurrence du test et des fonctionnalités des composants, de partage de ressources entre le test et les fonctions des composants, ce qui rend nécessaire la mise en place de mécanismes qui perturbent le moins possible les fonctions des composants.

Dans la suite de ce chapitre, la mise en œuvre du test des composants logiciels en vue d'un diagnostic en ligne de niveau système est analysée. Les principales contributions de nos travaux sont présentées.

V. Contributions

V.1. Niveau d'implémentation

Comme mentionné dans l'introduction, la plupart des solutions de tolérance aux fautes proposées dans le cadre des applications à base de composants logiciels ont recours à la réplication. Ainsi, l'approche que nous proposons se différencie de l'existant par l'utilisation de tests inter-composants en ligne.

L'étude des approches existantes de tolérance aux fautes par réplication, menée durant la thèse de T.-Q. Bui [25], a permis d'aboutir à une classification de ces approches en trois familles. Celles-ci se distinguent par le niveau d'implémentation des mécanismes de tolérance aux fautes mis en œuvre. Elles sont résumées ci-dessous (voir figure 1.2).

- Des approches de niveau *système* : où les mécanismes de tolérance aux fautes sont intégrés dans les couches les plus basses, ce qui permet l'accès à un certain nombre d'informations de niveau système [9].
- Des approches de niveau intermédiaire (*middleware ou intergiciel*) : appelées également approches par *intégration*, dont le principe de base est d'intégrer les mécanismes de tolérance aux fautes dans l'intergiciel ou le serveur de composants [58].
- Des approches de niveau applicatif :

- Les *services/librairies*, qui implémentent certains mécanismes de tolérance aux fautes, à charge de l'utilisateur de les appeler en les liant à son application [10].
- Les systèmes *réflexifs* ou par *héritage*, qui se placent au niveau du *langage*. Ils utilisent des mécanismes du langage, réflexivité ou héritage afin de lier les mécanismes de tolérance aux fautes et l'application [11] [12] [13].

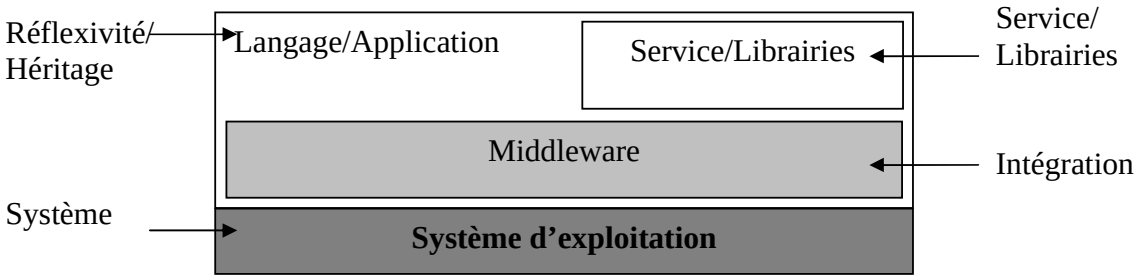

Figure 1.2. Différentes couches d'intégration de la tolérance aux fautes [25]

Divers critères ont été analysés dans la thèse de T.Q. Bui [25] pour permettre la sélection d'un niveau d'implémentation. Nous en retenons deux : la *transparence* et la *portabilité*. La transparence permet d'évaluer la non implication de l'utilisateur dans la mise en œuvre de la solution de tolérance aux fautes alors que la portabilité indique l'indépendance de la solution de tout support matériel, intergiciel ou environnement d'exécution. A l'évidence, les approches de niveau système et applicatif par services présentent un grand degré de transparence, ce qui est moins le cas pour les autres approches. Quant à la portabilité, elle est favorisée par les approches fondées sur des services, des librairies ou de niveau langage. L'approche service, qui satisfait aux deux critères retenus, répond le mieux à la mise en œuvre d'une solution de diagnostic à base de tests inter-composants. Dans le contexte de nos travaux, elle permet notamment l'automatisation des actions liées au test et au diagnostic, assurant ainsi un large degré de transparence à l'utilisateur. Enfin, elle nous permet également de nous affranchir d'un modèle de composant précis.

V.2. Le service DISCO (DIagnosis Service for Component-based applications)

V.2.1. Les tests inter-composants en ligne

La mise en place de ces tests a nécessité la résolution des problèmes suivants :

- ***L'implémentation des tests*** : un composant est considéré comme une boîte noire avec des interfaces fonctionnelles fournies et requises. Par conséquent, les fonctionnalités de test peuvent être considérées comme étant d'autres services que le composant fournit à son environnement (voir figure 1.3). Le composant testeur peut donc utiliser cette interface de la même façon que les autres interfaces fonctionnelles. Ainsi, une interface de test est intégrée dans les composants pour fournir des fonctionnalités de test [25] [59].

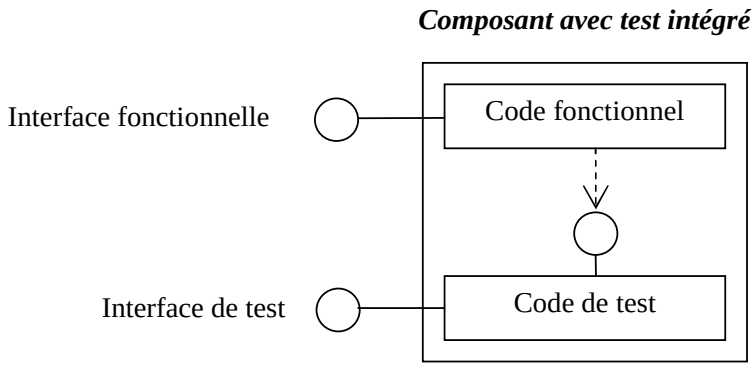

Figure 1-3. Composant avec test intégré [25]

Les principaux outils utilisés pour modéliser le fonctionnement d'un composant logiciel comportent des approches *formelles* telles que les *spécifications algébriques* et des approches *semi-formelles* parmi lesquelles nous retrouvons les *réseaux de Pétri*, des *langages spécifiques* mais plus communément les *machines à états finis*. Ce dernier formalisme nous a semblé particulièrement adapté en raison de sa simplicité et de la disponibilité d'outils et de nombreuses approches de test fondées sur ce modèle [60].

- ***Le contexte en ligne*** : Comme nous l'avons présenté dans le paragraphe IV.3. du Chapitre 1 sur les travaux de test en ligne existants, l'utilité de ceux-ci en vue du diagnostic et de la tolérance aux fautes n'a pas été clairement exploitée. Cette problématique a été abordée dans nos travaux sur deux axes spécifiques :
  - Le développement de tests inter-composants et leur exécution en prenant en compte la concurrence et les interférences possibles avec les fonctionnalités du composant,
  - L'analyse des résultats des tests pour localiser précisément les éléments défaillants, ce qui constitue la phase de diagnostic.

Contrairement à ce qui a été vu dans le test de déploiement, dans le cas du diagnostic en ligne, les tests de composants s'exécutent de manière concurrente avec les fonctionnalités des composants. Ainsi, un aspect important consiste à gérer les interférences entre les tests et les fonctionnalités normales du composant. Ces interférences concernent principalement les deux situations suivantes :

- La réception d'une demande de service fonctionnelle pendant le test, ou inversement, la réception d'une demande de test pendant l'exécution d'un service.
- La manipulation et la modification des données fonctionnelles du composant par le test.

Concernant la gestion des interférences entre le test et les fonctionnalités du composant, une idée intuitive, inspirée du test des composants matériels [61], consiste à exploiter les périodes d'oisiveté d'un composant. Ainsi, un composant oisif serait candidat à un test inter-composant, que ce soit dans le rôle du composant testeur ou dans celui du composant testé. Cependant, en pratique il n'est pas évident de détecter les composants oisifs. En effet, une

telle détection est étroitement liée au système d'exploitation sous-jacent, ce qui réduit la portabilité du test inter-composant. D'autres approches sont mentionnées dans la littérature, notamment dans [62]. Toutefois, à notre connaissance, ces approches n'ont pas fait l'objet d'une implémentation concrète. Nous pouvons mentionner :

- *Blocage du composant* : le composant est bloqué pendant le processus de test. Au cours de l'exécution d'un test, les demandes de test ou les requêtes fonctionnelles en provenance d'autres composants de l'application sont retardées jusqu'à la fin du processus de test. Il est nécessaire de restaurer les données persistantes du composant à leur état original après la fin du processus de test.
- *Abandon du processus de test* : le composant abandonne le processus de test. Quand un autre composant demande la fonctionnalité normale du composant testé, le processus de test est abandonné. Une fois abandonné, les données du composant doivent être restaurées.
- *Clonage du composant* : le composant est cloné par l'infrastructure d'exécution avant le démarrage du test. Le processus de test est alors exécuté sur le clone, et la fonction normale est exécutée sur le composant original. Avec cette solution, le fonctionnement normal de l'application n'est pas perturbé. Cependant, cette option peut être très coûteuse en consommation de ressources.
- *Session de test* : l'interface de test du composant fournit des méthodes qui permettent des sessions de test. Ces méthodes garantissent que les données de test et les données réelles ne sont pas mélangées durant le processus de test. Avec cette solution, le fonctionnement normal de l'application n'est pas perturbé mais une charge de travail supplémentaire pour le développeur de composants ou de l'application est inévitable.

Il est important de noter qu'aucune solution n'est optimale et que le choix d'une solution plutôt qu'une autre dépend du contexte considéré. L'analyse du principe de ces différentes stratégies nous amène à proposer la démarche suivante :

- Quand les applications considérées sont temps réel ou critiques, les demandes de test sont abandonnées au profit des appels fonctionnels. Il est toutefois nécessaire de garantir que tout composant est testé au moins une fois dans une période donnée, sinon l'apparition de fautes peut passer inaperçue et nuire au fonctionnement de l'application.
- Quand les applications ne sont pas critiques, il est envisageable d'interrompre momentanément un composant en vue de le tester. Pour ce type d'applications, la solution tient compte de la nature des opérations (*i.e.* lecture ou écriture) effectuées sur les données d'un composant lors de l'exécution d'un service :
  - Quand les fonctions du composant sous test sont de type lecture seule, le service est simplement bloqué jusqu'à la fin de l'exécution du test du composant.

- Quand les opérations sous test modifient des données, le recours à des sessions de test est plus adapté pour éviter l'utilisation des données effectives du composant lors du test.

V.2.2. Architecture du service DISCO

Une architecture de service de diagnostic, appelé DISCO (DIagnosis Service for software Components) a été proposée. Cette architecture est générale et indépendante des applications et de leurs modèles de composants. En outre, elle est flexible et s'adapte à l'évolution dynamique des applications. L'architecture proposée pour le service de diagnostic assure les opérations suivantes selon deux phases distinctes :

- Une phase précédant le déploiement des composants, durant laquelle les composants sont analysés en vue d'extraire un modèle d'états et les cas de tests à partir de leurs spécifications. C'est durant cette phase que les composants sont rendus testables par l'intégration d'une interface de tests.
- Une phase intervenant durant le déploiement des composants et qui consiste à regrouper les composants de l'application en groupes de diagnostic. L'objectif de cette phase est d'assurer une meilleure adaptation des tests et du diagnostic à la nature des composants sous test ainsi qu'une meilleure extensibilité du service.
- Enfin, une phase intervenant en ligne durant l'exécution et qui consiste à appliquer la stratégie de test choisie et à assurer le diagnostic des composants. Durant cette phase, le service DISCO gère également les ressources disponibles (mémoire, temps CPU).

Les travaux concernant cette architecture, les fonctions de base du service DISCO, les interfaces des composants et les interactions entre les composants ont été présentés dans les conférences internationales suivantes EFTS'07 [63], SIES'08 [64], Qualita'07 [65] et Qualita'09 [66]. Les aspects liés à l'analyse des composants en vue d'une insertion automatique de l'interface de test ont été approfondis dans le stage de Master (M2R) de N.-Q. Tran [59].

V.2.3. Implémentation et validation

Pour valider ces travaux, des analyses et évaluations expérimentales du service DISCO sur différents modèles d'applications en vue de déterminer le surcoût dû au diagnostic et d'évaluer ses performances ont été menées.

Les réalisations pratiques ont concerné l'implémentation d'une plate-forme d'expérimentation qui permet de modifier de manière souple les paramètres de diagnostic, par exemple la durée de test, le nombre de composants, *etc*., pour analyser les performances du service DISCO. Le service DISCO a été porté sur un cas d'étude industriel – le système de gestion à grande échelle de données de capteurs hétérogènes développé dans le cadre d'une collaboration entre le laboratoire LIG et France Télécom[2]. Ce travail a donné lieu à une publication dans Depend'09 [67][3].

[2] Il s'agit du système SStreamWare développé dans le cadre d'une collaboration entre le laboratoire LIG et la société France Telecom.
[3] L'article correspondant est inclus en Annexe B de ce mémoire.

VI. Bilan et perspectives

VI.1. Bilan

Ce chapitre a résumé les travaux menés dans le domaine du test et du diagnostic en ligne de composants logiciels. Ces travaux nous ont amenés à étudier l'existant dans divers domaines, en particulier :

- Les différents modèles de composants logiciels ;
- Les approches de test et plus spécifiquement le test d'intégration des composants logiciels ainsi que l'existant au niveau du test en ligne de ces composants ;
- Les approches de sûreté de fonctionnement proposées pour assurer la continuité de service dans les applications développées à partir de composants logiciels ainsi que les approches de diagnostic système existantes.

A partir de ces différents éléments, une solution de diagnostic en ligne fondée sur des tests inter-composants logiciels a été développée et réalisée sous la forme d'un service, appelé DISCO. A travers ce travail, nos principales contributions ont concerné essentiellement l'adaptation du test intégré de composants logiciels à une utilisation en ligne en reportant les principales actions (intégration du test dans les composants, génération et exécution des cas de tests, diagnostic, *etc*.) au niveau de l'utilisateur pour une meilleure efficacité des tests et une indépendance renforcée de l'utilisateur vis-à-vis du développeur. Ceci a nécessité un travail d'automatisation en vue d'aider l'utilisateur dans sa démarche de test et de diagnostic en ligne.

Ces travaux se sont déroulés dans le cadre de la thèse de T.-Q. Bui [25] et ont concerné le projet FITT « Détection et localisation de composants embarqués défaillants » et le projet européen EURNEX. Après avoir été portés sur un système de gestion de données de capteurs, ils se sont également poursuivis dans le cadre du projet BQR ARTeco avec le laboratoire LIG et ont servi de base au projet CEDRE avec l'Université Libanaise ainsi que dans le cadre de la thèse de D. Hamdan.

VI.2. Perspectives : Unification matérielle/logicielle

Le travail présenté dans ce chapitre s'appuie sur des tests inter-composants et le diagnostic en ligne pour des applications à base de composants logiciels. Il s'inspire en cela des travaux de diagnostic en ligne de composants matériels (processeurs) interconnectés par un réseau.

L'analogie de nos travaux avec ceux relatifs à *l'autotest logiciel du matériel* (*Software-Based Self-Test*) [68] [69] ainsi que ceux relatifs au *diagnostic au niveau logiciel des défauts matériels* [70] nous motive à analyser de plus près l'extension de notre approche aux cas des systèmes matériels/logiciels. Ceci suppose une analyse préalable des interactions entre les composants matériels et les composants logiciels d'un système ainsi que le développement d'une approche de diagnostic globale qui exploite les tests de composants logiciels ainsi que

l'autotest logiciel de composants matériels. Un modèle unifié de composants logiciels/matériels serait un élément important de réussite d'une telle approche.

Les paragraphes ci-dessous résument nos réflexions préliminaires dans ce sens.

VI.2.1. Composants matériels

Par analogie avec les composants logiciels tels qu'on les connaît aujourd'hui, il n'existe pas, à notre connaissance, de modèles de composants matériels définis de manière formelle. Cependant, il est aisé de constater que le logiciel est de plus en plus présent dans le développement matériel à travers des descriptions logicielles des systèmes matériels. Ces descriptions se présentent souvent sous la forme de modules (VHDL, Verilog) ou d'objets (SystemC) et fournissent des interfaces permettant d'exploiter les fonctionnalités du composant matériel, tout comme le font les modèles de composants logiciels. Par ailleurs, un composant matériel physique s'utilise également à travers une interface bien définie comportant des ports ou des sondes à fonctionnalités précises (*cf.* la norme JTAG pour le test et le diagnostic du matériel). C'est en particulier en exploitant des sondes et des ports de communication précis que le test et le diagnostic du matériel ont été établis, d'où la question de l'extension des approches existantes pour le matériel aux composants logiciels. Ainsi, comme nous l'avons vu dans l'état de l'art sur le test de composants logiciels, celui-ci s'inspire dans une large mesure, tant au niveau de la terminologie que de la mise en œuvre, des travaux existants pour le matériel.

Le test des systèmes matériels et sur puce se fait classiquement par un testeur externe ou par autotest matériel. Cependant, les techniques reposant sur les équipements de test externes (*Automatic Test Equipment ATE*) se dirigent vers une impasse à cause de l'inaccessibilité des nœuds internes depuis l'extérieur de la puce et l'utilisation de mémoires de plus en plus larges. Face aux problèmes posés par le test depuis l'extérieur de la puce, une solution est d'intégrer davantage de fonctionnalités consacrées au test, fournissant ainsi des puces de plus en plus auto-testables. Cependant, les techniques d'autotest matériel introduisent de la circuiterie supplémentaire dédiée au test, et ces circuits souffrent de ce matériel annexé.

Avec l'avènement des systèmes sur puce (SoC) qui permettent d'exécuter du logiciel embarqué, la technique de l'autotest logiciel est en plein essor. Il s'agit de réutiliser les capacités des SoC à exécuter du logiciel embarqué, à des fins d'autotest. C'est une approche non intrusive, de bas coût, flexible et programmable. De telles approches qui prennent compte simultanément du logiciel et du matériel présents dans un système pourraient être explorées à l'avenir dans l'objectif d'approfondir et d'améliorer l'efficacité des méthodes que nous proposons.

VI.2.2. Vers une vision uniforme des composants logiciels et matériels

Certains auteurs ont poussé l'analogie au point de proposer un modèle uniforme de composant, pouvant convenir tant aux composants logiciels qu'aux composants matériels, à l'image du modèle Ucomp décrit dans [71]. La motivation des auteurs est venue de la constatation que les systèmes embarqués actuels comportent des unités matérielles couplées avec des composants logiciels, souvent répartis sur différents nœuds et la quasi-absence de

modèles permettant une coopération entre composants matériels et logiciels. Les auteurs notent toutefois que les standards OPC [22] pour le développement de logiciels dans le domaine du contrôle industriel et AUTOSAR [21] qui concerne le développement de logiciels embarqués pour l'industrie automobile permettent un certain degré de coopération entre les composants matériels et les composants logiciels. Mais ces modèles s'appliquent à des domaines précis. L'objectif de leur travail a donc été d'offrir un modèle de composant et un framework qui permettent de traiter le matériel et le logiciel de manière uniforme durant les phases de conception et d'utilisation du système. Ainsi le modèle proposé, UComp, comporte les spécifications d'éléments architecturaux pour des composants logiciels et matériels. Il définit deux types de composants : les « device components » qui « enveloppent » un composant matériel ou réseau virtuel et présentent leurs fonctionnalités en tant que composant et les « composants logiciels » dont la fonctionnalité est entièrement définie sous forme de code et qui ne sont pas liés à un dispositif matériel.

Un tel travail indique le besoin formulé par les développeurs de systèmes embarqués d'uniformiser la vue globale d'un système comportant des composants logiciels et matériels. En étendant l'approche de diagnostic à base de tests inter-composants aux composants logiciels, nos travaux rejoignent ces préoccupations et peuvent servir de socle à une approche unifiée de test et de diagnostic en ligne. Ces réflexions orienteront nos travaux futurs dans cet axe de recherche[4].

Le chapitre qui suit décrit les travaux que nous avons menés sur les systèmes RFID. Il s'agit de systèmes hétérogènes, comprenant du matériel, du logiciel et dont le fonctionnement est dépendant de l'environnement. Ces travaux représentent un premier pas vers une approche de test et de diagnostic globale.

[4] La thèse de B. Sahbi, qui se déroule dans le cadre d'une convention CIFRE entre le laboratoire LCIS et l'entreprise Defacto Technologies à Moirans a débuté au mois de mars 2014 et abordera ces préoccupations.

Chapitre 2 TEST ET DIAGNOSTIC EN LIGNE DES SYSTEMES RFID

I. Positionnement de l'étude

I.1. Introduction

On assiste de nos jours à une utilisation croissante des systèmes RFID (RadioFrequeny IDentification) dans divers domaines d'application (logistique, systèmes de production, inventaires, traçabilité, *etc*.) Les échanges électroniques constituent un facteur de renforcement de la compétitivité des entreprises dans un contexte de concurrence internationale et de mondialisation des échanges et des flux matériels. L'observation des gains de productivité réalisés avec le code-barres et la standardisation qui a accompagné son déploiement laissent augurer des gains encore plus significatifs avec les étiquettes électroniques[5] dans les applications mettant en œuvre de gros volumes : suivi et optimisation des opérations de production, optimisation des stocks, lutte contre le vol et la contrefaçon, optimisation de la chaîne logistique, amélioration du service au consommateur, *etc*.

De par leurs diverses applications, les systèmes RFID font partie de ce qui est connu comme étant *l'Internet des Objets* (*Internet of things*), large sujet de recherche englobant en outre les réseaux de capteurs et les appareils mobiles (téléphones portables, *etc*.) [72].

Certaines de ces applications présentent un caractère critique, par exemple le respect de la chaîne du froid lors de l'acheminement de denrées alimentaires ou dans le cas de systèmes médicaux intégrés ou non au corps humain. Or, la sensibilité des systèmes RFID à leur environnement, notamment aux perturbations électromagnétiques ou à la présence d'obstacles, les rend particulièrement vulnérables. De plus, en raison du nombre important d'éléments (étiquettes, lecteurs) mis en œuvre dans de tels systèmes, des comportements erronés peuvent survenir suite à des fautes dans les divers éléments constituant le système. D'où l'importance et la nécessité de traiter le problème de la sûreté de fonctionnement et de la tolérance aux fautes dans le but de rendre ces systèmes plus robustes.

Les systèmes RFID modernes sont des systèmes complexes comportant des éléments logiciels et d'autres matériels pouvant être numériques ou analogiques. Par conséquent, l'étude de leur sûreté de fonctionnement est une problématique assez complexe faisant

[5] Dans ce document, les mots « étiquette » et « tag » sont utilisés indifféremment.

intervenir divers domaines de compétences (informatique, électronique, télécommunications, *etc*.) La problématique « sûreté de fonctionnement des systèmes RFID» a été peu étudiée au profit de techniques de test de production spécifiques aux étiquettes RFID et surtout des problématiques de sécurité-immunité (ou sécurité) qui s'intéressent essentiellement à l'intégrité et la confidentialité des données échangées.

Ces différents points ont conduit à définir la problématique suivante : assurer la sûreté de fonctionnement d'un *système réparti*, combinant du *matériel* et du *logiciel* d'une part, et d'autre part reposant sur des *composants matériels nombreux*, de *faible coût* et au fonctionnement « *incertain* ».

Ce travail a eu pour cadre le projet ANR SafeRFID [73] et a concerné les thèses de Doctorat de G. Fritz [74] et R. Kheddam [75]. Il s'est déroulé en étroite collaboration avec la plate-forme RFTLab de Grenoble INP, située à Valence, en particulier pour tous les aspects liés aux normes RFID et aux expérimentations pratiques.

I.2. Méthodologie de travail

Pour mener à bien ces recherches dans un domaine relativement nouveau, il a été nécessaire d'analyser les différents composants d'un système RFID pour déterminer les points de vulnérabilité à traiter par la définition de modèles de fautes et la proposition de méthodes de sûreté de fonctionnement appropriées. Nous avons opté pour deux axes de travail (voir figure II.1) :

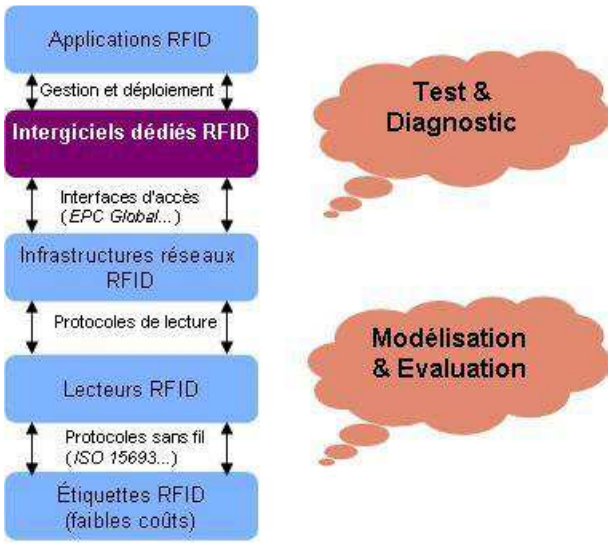

Figure II.1. Portée du projet SafeRFID

- tout d'abord, l'étude du lien et des interactions entre les lecteurs et les étiquettes RFID dans le but d'identifier toute dégradation de la qualité de la communication tag-lecteur,

- par la suite, l'étude du middleware RFID et du protocole LLRP (Low Level Reader Protocol) qui assure la communication entre le middleware et les lecteurs dans le but de proposer une approche de test et de diagnostic en ligne qui prend en compte le système RFID dans sa totalité.

Dans le cadre de nos activités de recherche, ce travail constitue un premier cas d'étude significatif de la prise en compte d'un système complet et des principales interactions entre ses composants pour assurer sa sûreté de fonctionnement. La nature des systèmes informatiques actuels et ceux à venir, comportant notamment une imbrication de composants de natures diverses et variées, devrait orienter les travaux de la sûreté de fonctionnement vers des approches globales, comme celle développée ici.

Lors de nos travaux d'analyse, nous avons eu recours à des outils classiques de la sûreté de fonctionnement, en particulier l'AMDE (Analyse des Modes de Défaillances et de leurs Effets) [76] qui nous a permis d'identifier les faiblesses tant au niveau du couple lecteur-tag qu'au niveau de la communication entre le middleware et les lecteurs.

La diversité des normes et des équipements (lecteurs, étiquettes de différentes catégories) a nécessité un affinement de nos choix en vue de définir un environnement de travail précis, notamment en ce qui concerne le type de lecteurs et d'étiquettes étudiés et les protocoles de communication considérés.

Ainsi, nous nous intéressons principalement à une technologie : la technologie UHF (Ultra Hautes Fréquences)[6]. En effet, cette technologie se retrouve largement dans des applications nécessitant un fonctionnement sûr, comme par exemple les applications critiques liées au nucléaire, au ferroviaire ou encore à l'aéronautique ; ou les applications de logistique, mettant en œuvre une grande quantité d'équipements (lecteurs, étiquettes) répartis géographiquement pour lesquelles les interventions humaines, suite à une panne, seraient difficiles et coûteuses en temps et en argent.

Ces recherches ont permis le développement d'un ensemble de méthodes et d'approches pour améliorer la sûreté de fonctionnement des systèmes RFID. En particulier, les résultats suivants ont été obtenus

Axe modélisation et évaluation : le travail mené dans cet axe a abouti à la proposition des éléments suivants :

- Une approche de *test statistique* des tags RFID dans le but de détecter les défauts de lectures. Cette approche, appelée méthode *Profil*, est une approche de test empirique fondée sur l'observation et l'analyse des taux de lectures des tags dans un système RFID [77].
- Le développement d'un simulateur appelé SERFID pour la *simulation des systèmes RFID HF et UHF* [78]. Ce simulateur a notamment été utilisé pour valider l'approche de test Profil.

[6] La technologie HF (Hautes Fréquences) a également été étudiée dans le cadre d'une collaboration avec l'entreprise Schneider.

Axe test et diagnostic : à l'instar des méthodes existantes, le test statistique proposé ne permet pas de localiser l'origine des problèmes observés. Aussi, avons-nous complété ce test au niveau du middleware par un processus de *diagnostic probabiliste* dont le principe de base est issu de la littérature et que nous avons largement adapté au cas des systèmes RFID. Cet axe a abouti aux propositions suivantes :

- La spécification et le développement d'un prototype de middleware RFID appelé SafeRFID-MW. Ce middleware est capable de diagnostiquer les lecteurs et les tags défaillants dans un système RFID selon une approche de *diagnostic probabiliste*. Ce diagnostic implémenté sous la forme d'un algorithme appelé RFID_diagAlgo, se fonde sur les résultats des tests effectués au niveau des lecteurs (par exemple, par la méthode Profil introduite ci-dessus ou tout autre méthode indiquant les performances des lecteurs) [79].
- Le diagnostic proposé est également complété par une analyse fine des fichiers de logs des échanges entre le middleware et les lecteurs RFID en vue de localiser précisément l'origine des dysfonctionnements observés [80] [81].

La suite de ce chapitre introduit les principaux éléments qui constituent un système RFID. Cette description est imitée aux aspects nécessaires à la compréhension des solutions de sûreté de fonctionnement développées. Par la suite, ce chapitre décrit les aspects novateurs de nos travaux sur la sûreté de fonctionnement des systèmes RFID, en particulier les approches de test et de diagnostic développées

II. Description des systèmes RFID

II.1. Principaux composants des systèmes RFID

Le principe de fonctionnement d'un système RFID est simple. Il s'agit d'échanger des informations numériques à moyenne distance par radiofréquences, ces informations devant être contenues sur un support très petit et de très faible coût. Un système RFID comprend deux composants de base :

- Un élément communément appelé *transpondeur*, *étiquette communicante* ou *tag* qui se compose d'une antenne et d'une puce électronique contenant des informations, des capteurs, *etc*. C'est l'élément associé à l'objet à identifier[7].
- Un appareil de lecture appelé aussi *interrogateur*, *station de base* ou *lecteur* qui permet d'inventorier[8] les étiquettes présentes dans son champ, de lire et d'écrire des informations numériques dans la puce du transpondeur.

[7] Chaque étiquette comprend un identifiant unique qui est aussi l'identifiant de l'objet tracé. Dans la norme EPCglobal qui nous intéresse, cet identifiant est appelé code EPC (Electronic Product Code).

[8] Un *inventaire* est une séquence de commandes qui permet à un lecteur RFID d'inventorier l'ensemble des tags présents dans son champ. Un rapport d'inventaire est produit par le lecteur et transmis au middleware (ou au système hôte). En pratique, un *inventaire applicatif* est constitué de plusieurs *inventaires simples* afin d'assurer la lecture d'un maximum de tags présents.

Dans la technologie UHF, une classification courante des étiquettes concerne leur mode de communication [82]. Ainsi, les étiquettes *passives* sont celles qui utilisent l'onde émise par le lecteur afin de lui envoyer un message. Les étiquettes *actives* possèdent leur propre émetteur ce qui leur permet d'atteindre des distances de communication plus importantes que dans le cas des étiquettes passives. Cependant, cela demande généralement une source d'énergie propre. On rencontre également dans la littérature des étiquettes *semi-passives* ou *semi-actives*. Dans ce cas il s'agit d'étiquettes passives munies d'une batterie.

Dans les applications qui nous intéressent (logistique, inventaire d'équipement, traçabilité, *etc*.), ce sont les étiquettes passives qui sont le plus couramment utilisées en raison de leur coût réduit. Or il s'agit d'une catégorie d'étiquettes peu robustes. Ceci s'explique par leur coût peu élevé mais motive également la recherche de solution adaptées pour les rendre plus robustes. De ce fait, nous nous intéressons à l'étude de ce type d'étiquettes.

Les différents systèmes RFID utilisés dans les systèmes de production sont généralement constitués de réseaux locaux (regroupement de plusieurs lecteurs) et globaux (regroupement de plusieurs réseaux locaux) ainsi que d'applications utilisant les données collectées et sauvegardées dans une base de données (voir figure 2.2). Dans ce contexte, les intergiciels RFID ou middlewares RFID permettent de transférer les données capturées par les lecteurs vers une application logicielle. Ces intergiciels constituent le lien entre le monde matériel et le monde logiciel et assurent les tâches suivantes :

- le contrôle des lecteurs,
- le contrôle des données (capture, filtrage et routage),
- la transmission des données aux applications.

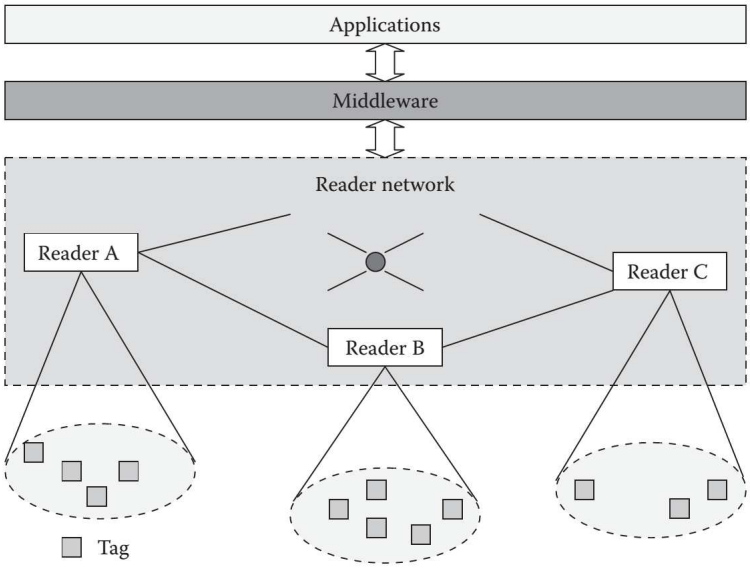

Figure 2.2. Architecture d'un système RFID distribué [75]

Depuis quelques années, les besoins en middlewares adaptés à la RFID augmentent de manière importante, comme le montrent la plupart des études de marché [83] ainsi que les

nombreux travaux ayant abouti au développement de middlewares capables de répondre à ces besoins particuliers, notamment la gestion de quantités importantes de données [84] [85] [86] [87] [88] [89] [90] [91]. Cependant, les applications RFID sont par nature déployées sur des équipements variés (équipements industriels, domestiques, médicaux, *etc.*) dont la fiabilité n'est pas toujours garantie et utilisées dans des environnements parfois défavorables à leur fonctionnement correct (champs électromagnétiques dans diverses applications telles que les applications ferroviaires par exemple, *etc.*). Ceci fait de la sûreté de fonctionnement de ces applications un enjeu important qui doit être pris en compte dans le développement des middlewares.

II. 2. Les fonctions d'un middleware RFID

Les principales fonctions d'un middleware RFID sont résumées ci-dessous (voir figure 2.3) [75] [92] :

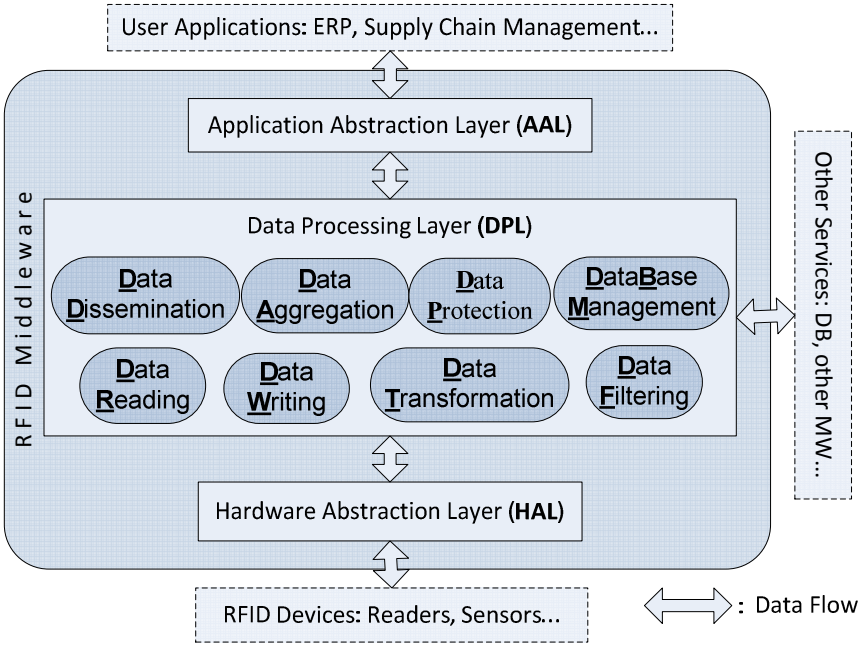

Figure 2.3. Architecture en couches d'un middleware RFID [75]

Dissémination des données RFID : de manière générale, les données RFID sont partagées entre plusieurs applications pouvant appartenir à plusieurs partenaires métier. Cette fonction a pour rôle de garantir que toute application ayant manifesté un intérêt pour une catégorie de données RFID reçoit bien les données attendues.

Agrégation des données RFID : les systèmes RFID engendrent de grandes quantités de données avec de nombreuses redondances. Le middleware se charge alors d'agréger cette grande masse de données en vue de ne transmettre que les informations utiles aux applications, en supprimant les redondances jugées superflues. Il est toutefois intéressant de noter ici que ces redondances jugées inutiles du point de vue des applications, peuvent être exploitées de manière efficace dans le cadre d'une solution de sûreté de fonctionnement,

comme cela est le cas dans la méthode de test développée dans nos travaux, appelée la méthode Profil et qui sera détaillée dans la suite de ce chapitre (*cf.* Chapitre 2, section IV.2).

Filtrage des données RFID : la plupart des applications RFID nécessitent des données filtrées selon certains critères qui peuvent être : l'identité du lecteur, l'identité du tag, l'identité d'une antenne de lecteur, *etc*. Pour répondre à ces besoins, les middlewares RFID mettent en place un filtrage des données selon le ou les critères définis par l'application.

Interprétation des données RFID : l'objectif ici est de déduire un événement métier en interprétant un événement RFID et de fournir ainsi à l'application une information directement exploitable.

Les différents middlewares RFID existants réalisent l'ensemble de ces fonctions en respectant certains normes et standards. Certains middlewares sont compatibles avec plusieurs standards alors que d'autres se limitent à un standard donné. Cependant, les différents standards internationaux (ISO 15693, ISO 14443, ISO 18000–6 A/B, EPCglobal, *etc*.) se focalisent principalement sur les caractéristiques architecturales, technologiques et fonctionnelles de ces systèmes : le niveau de puissance des lecteurs, les protocoles de communication RFID, les intervalles radiofréquences autorisés, les temps de réponse maximums, *etc*. La sûreté de fonctionnement y est souvent mentionnée comme étant un élément clé du développement des systèmes RFID, mais sans la formalisation d'une solution spécifique. En pratique, la solution la plus largement mise en œuvre au niveau middleware pour assurer un fonctionnement correct des systèmes RFID consiste à surveiller les différents lecteurs connectés au système et à ignorer toute donnée en provenance d'un lecteur jugé défaillant. Quelques-unes des approches existantes sont mentionnées dans la suite de ce chapitre (*cf.* Chapitre 2, section IV.1).

Les travaux conduits selon les deux axes de recherche fixés ci-dessus sont décrits ci-dessous dans les paragraphes III (axe « *modélisation et évaluation* ») et IV et V (axe « *test et diagnostic* ») respectivement.

III. Analyse, modélisation et simulation des systèmes RFID

L'AMDE [76] est une méthode assez ancienne puisqu'elle date des années 60. Elle est cependant très utilisée dans les études de sûreté de fonctionnement. Il s'agit d'une méthode inductive qui permet l'analyse d'un système en se basant sur sa structure ou en déterminant ses principales fonctions et ses composants clés. Une part importante de cette étude consiste à déterminer les dysfonctionnements probables qui peuvent survenir dans le système étudié, leurs causes ainsi que les méthodes pouvant les traiter. En raison du manque de travaux antérieurs sur la sûreté de fonctionnement des systèmes RFID, une telle étude nous a semblé incontournable afin d'identifier les défauts susceptibles de perturber le fonctionnement du système et d'orienter nos travaux vers la recherche de solutions adaptées. Nous avons mené cette étude séparément pour les deux niveaux qui nous intéressent dans un système RFID, c'est-à-dire le couple tag-lecteur et la communication entre le middleware et les lecteurs. En effet, la communication tag-lecteur est un point de vulnérabilité évident de ce type de

systèmes alors que le middleware a été identifié comme le support de la solution globale visée par nos travaux.

Ce paragraphe présente les principales conclusions de l'AMDE spécifique au couple tag-lecteur. Les détails de cette étude sont présentés dans la thèse de G. Fritz [74].

III.1. AMDE du système tag-lecteur

Pour cette AMDE, deux découpages du système tag-lecteur ont été proposés permettant ainsi d'avoir deux points de vue sur le système formé par le couple tag-lecteur. La communication étant la fonction principale de ce système, elle est au cœur de ces deux découpages. Le premier découpage est opéré par couches selon le modèle ISO/OSI [93] mettant en avant le déroulement de la communication entre le tag et le lecteur. Le second découpage est un découpage fonctionnel, fondé sur les principaux sous-composants fonctionnels du tag et du lecteur et leurs rôles respectifs (voir figure 2.4.).

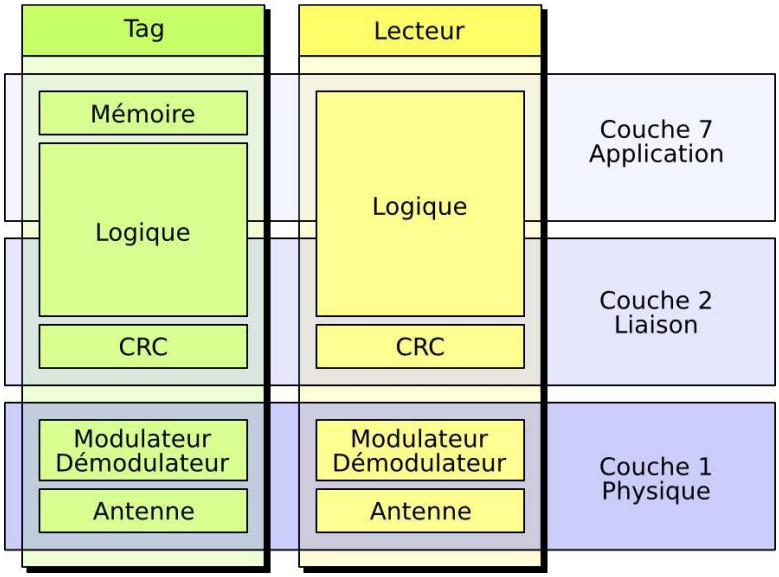

Figure 2.4. Relation entre le modèle OSI simplifié et les modèles comportementaux du tag et du lecteur [74]

Mener à bien cette AMDE a également nécessité une étude approfondie des deux normes les plus courantes en RFID : la norme EPCglobal Class1 Gen2 pour la RFID UHF et la norme ISO 18000-3 pour la RFID HF, avec toutefois un intérêt particulier pour la norme EPCglobal Class 1 Gen 2 qui concerne directement les applications visées par nos travaux. L'étude de ces normes a permis de prendre en compte les éléments qui y sont prévus et qui répondent à l'analyse de sûreté de fonctionnement.

Comme la communication entre un tag et un lecteur est relativement simple, seulement trois couches du modèle ISO/OSI ont été retenues :

- La couche 7 ou couche application, qui gère les actions à réaliser : envoi de requêtes et de réponses d'identification, *etc*.
- La couche 2 ou couche liaison, qui prépare le message de la couche application pour le transmettre à travers la couche physique (ajout d'un CRC, *etc*.) Cette couche assure également la vérification des messages en provenance de la couche physique avant leur livraison à la couche application (vérification du CRC, *etc*.)
- La couche 1 ou couche physique qui assure la transmission des données binaires en assurant la modulation, l'envoi du signal sur le médium, la récupération du signal présent sur le médium et sa démodulation.

Le découpage fonctionnel met en avant les fonctions classiques des systèmes communicants : antenne, modulateur, démodulateur, calculateur de CRC, logique de contrôle mémoire, composant de récupération d'énergie pour le tag et composant de sélection d'antenne pour le lecteur.

A partir de ces deux découpages, et pour chaque composant défini, les modes de défaillances possibles ainsi que leurs causes éventuelles ont été identifiés.

A titre d'illustration, un mode de défaillance identifié dans la communication tag-lecteur est la « non réception des signaux par un tag » [74]. A ce mode de défaillance, ont été associées plusieurs causes possibles :

- Le tag est en dehors du champ du lecteur alors qu'il devrait y être.
- Des perturbations dues à l'environnement extérieur comme par exemple un choc ayant abîmé l'antenne du tag, ou des conditions de stockage ou d'utilisation hostiles au bon fonctionnement du tag.
- Des défaillances internes telles que des mauvaises jonctions au sein du tag ou des défaillances de ses composants.

Bien que longue et fastidieuse, une telle étude a l'avantage de fournir une connaissance plus approfondie du fonctionnement des systèmes cibles et de mieux connaître leurs faiblesses. Dans le cadre de nos travaux, elle a permis en outre une meilleure compréhension du fonctionnement des systèmes RFID et nous a permis de proposer un modèle de fautes pour le couple tag-lecteur décrit ci-après[9].

III.2. Modèles de fautes pour le couple tag-lecteur

Les modes de défaillances déterminés suite à l'étude AMDE et les principales causes qui ont été identifiées ont permis de converger vers des modèles de fautes représentatifs des dysfonctionnement les plus probables lors de la communication entre un lecteur et plusieurs tags RFID. En particulier, les trois modèles de fautes décrits ci-dessous, permettent de couvrir plusieurs modes de défaillances.

- La désactivation, puis la réactivation du lien de communication entre le tag et le lecteur. Ce modèle de faute permet de représenter la sortie d'un tag du champ d'un

[9] Le support et l'expertise fournis par le RFTLab ainsi qu'un travail de recherche commun avec l'entreprise Schneider ont permis de valider cette étude AMDE, en apportant le regard d'experts du domaine.

lecteur, ainsi que la non-réception ou la non-émission des signaux par un ou plusieurs tags ou lecteurs.

- L'impossibilité de communiquer entre un lecteur et un ensemble de tags alors que ces derniers sont dans le champ du lecteur (et sont donc alimentés). Ce modèle de fautes permet de prendre en compte les interférences de type compatibilité électromagnétique (CEM) qui peuvent survenir au niveau d'un ensemble de lecteurs et de tags. Ces problèmes empêchent alors la communication de se dérouler normalement malgré la présence de l'alimentation.
- La diminution de la qualité du lien tag-lecteur. Ce modèle correspond à l'augmentation du nombre d'erreurs binaires[10] lors de la communication entraînant une mauvaise émission ou réception des signaux par un ou plusieurs tags ou lecteurs.

Ces modèles de fautes permettent également de représenter le dysfonctionnement de plusieurs sous-composants des tags et des lecteurs impliqués dans la communication (modulateur, démodulateur, antenne, *etc*.).

Leur mise en œuvre pratique dans le cadre d'expérimentation *in situ* est toutefois complexe et difficile. Or, l'objectif de notre travail étant de tester le comportement d'un système RFID en y injectant des fautes afin de pouvoir proposer des méthodes de sûreté de fonctionnement adaptées, le recours au développement d'un simulateur nous a semblé nécessaire pour pouvoir mener nos recherches de manière plus approfondie. La spécification et le développement d'un simulateur pour les systèmes RFID se sont déroulés dans le cadre du projet SafeRFID et plus précisément durant la thèse de G. Fritz. Ils sont décrits ci-dessous.

III.3. Simulation des systèmes RFID

La simulation des systèmes RFID a été abordée dans la littérature sous trois niveaux principalement : tag, lecteur et lien tag-lecteur. Or, dans notre démarche, c'est l'ensemble du système RFID qui est visé par nos travaux. Ce sont par conséquent les simulateurs qui prennent en compte la globalité d'un système RFID qui nous intéressent. Quelques exemples de tels simulateurs sont présents dans la littérature, mais nous verrons qu'ils ne couvrent pas la sûreté de fonctionnement.

III.3.1. Simulateurs existants

Les simulateurs disponibles pour les systèmes RFID traitent en général seulement une partie de ces systèmes. Les différentes catégories identifiées de simulateurs sont :

- Des simulateurs qui ont pour objet l'étude du fonctionnement des tags et la validation de nouvelles fonctionnalités qui leur sont liées [94] [95].
- Des simulateurs qui modélisent le fonctionnement des lecteurs RFID par la génération de données à destination des middlewares. Ces données représentent les tags lus et servent à modéliser et analyser les flux générés par les lecteurs ainsi qu'à la validation des middlewares, de la charge des systèmes, *etc*. [96] [97].

[10] Ce nombre dépend du taux d'erreurs binaires qui représente le pourcentage d'inversions de bits qui se produisent lors de la transmission de flux de données numériques.

- Des simulateurs qui permettent d'étudier le lien entre le tag et le lecteur en analysant le protocole logique ou le lien physique entre le tag et le lecteur. Ces simulateurs permettent de définir les paramètres du protocole tag-lecteur [98] [99].
- Quelques simulateurs publiés ont pour objectif d'étudier les systèmes RFID dans leur globalité [100] [101] [102]. Ce type de simulateurs est le plus enclin à nous intéresser en raison de la prise en compte de l'ensemble des éléments présents dans un système RFID.

Nous retrouvons dans cette dernière catégorie, le simulateur présenté dans [100] qui combine l'utilisation de SystemC (comme librairie du langage C++) et de l'environnement de simulation Matlab Simulink® pour simuler respectivement la couche liaison et la couche physique. La simulation de la couche physique se fait ici sur la base des résultats de simulation obtenus pour la couche liaison, rendant le processus de simulation assez lourd et relativement limité puisqu'il n'est pas possible d'observer par simulation simultanément la couche liaison et la couche physique.

Le second simulateur global pour les systèmes RFID est présenté dans [101] [102] et est connu sous le nom de RFIDSim. Ce simulateur permet de simuler trois éléments essentiels d'un système RFID : le tag, le lecteur et le protocole de communication tag-lecteur. Ce simulateur implémente la norme EPCglobal Class 1 Gen 2 et considère uniquement les systèmes RFID UHF. Bien qu'intéressant, ce simulateur nous a semblé incomplet pour mener une étude de sûreté de fonctionnement car les aspects d'injection de fautes et d'analyse de robustesse n'y sont pas prévus. De même, il ne comprend pas les aspects d'interaction avec le middleware qui sont au cœur de notre travail.

Par conséquent, les simulateurs existants sont relativement limités pour l'étude des aspects liés à la robustesse des systèmes RFID. Nous avons donc opté pour le développement de notre propre simulateur dans l'objectif de :

- tenir compte plus précisément du fonctionnement des différents composants matériels,
- modéliser l'interaction avec le middleware,
- permettre l'injection de fautes selon les modèles de fautes issus de l'étude AMDE,
- évaluer les approches de test des systèmes RFID.

III.3.2. Développement du simulateur SERFID (Simulation et Evaluation des systèmes RFID)

Plusieurs options de développement se présentent lorsque l'on projette de concevoir un simulateur. Afin de déterminer celles qui sont le mieux adaptées à l'analyse de la robustesse des systèmes RFID, il est très important de définir clairement les objectifs attendus du simulateur. Dans notre cas, ces objectifs peuvent se résumer comme suit :

- *Simulation fonctionnelle* de l'ensemble d'un système RFID : tag, lecteur, protocole tag-lecteur, middleware. L'existence de middlewares généraux permet de se limiter à la simulation de la partie matérielle : tag-lecteur et protocole tag-lecteur, et envisager par la suite une interface avec un middleware RFID. Ceci est d'autant plus vrai dans le

contexte de nos travaux que le développement d'un middleware RFID intégrant la solution de diagnostic proposée est l'un des objectifs du deuxième axe de recherche (*cf*. Chapitre 2, section V).

- Récupérer des informations sur le *comportement temporel* du système afin de pouvoir extrapoler au cas d'un système réel et évaluer l'impact des approches de test sur les performances d'un système réel.
- Modélisation des *effets aléatoires*, caractéristiques du fonctionnement des systèmes RFID.
- *Pourvoir permuter* facilement des composants pour étudier par exemple diverses normes liées aux systèmes RFID, en particulier les fonctionnements HF et UHF.

Plusieurs options de développement du simulateur SERFID ont été analysées dans le cadre de la thèse de G. Fritz [74]. Nous citons notamment l'utilisation des éléments suivants :

- un environnement de simulation comme Matlab Simulink®,
- des langages de descriptions matérielles (VHDL, Verilog),
- des langages de programmation classique (Java, C, C++),
- des bibliothèques spécifiques de simulation à événements discrets ou de simulation de signaux analogiques (JiST, SystemC, SystemC-AMS).

Un critère de décision important concerne la possibilité de modéliser de manière conjointe des composants matériels et logiciels. Il est en outre important de pouvoir tenir compte des signaux analogiques et numériques. A ce titre, la bibliothèque SystemC nous a semblé la mieux adaptée. De plus, son utilisation se fait conjointement avec le langage C++, qui de par son aspect orienté-objet, permet l'instanciation de plusieurs composants, répondant dans notre cas à la nécessité d'avoir plusieurs tags et plusieurs liens tags-lecteurs pour l'étude des systèmes RFID complexes.

Modélisation d'un système RFID

La modélisation choisie est fonctionnelle car nous souhaitons observer et étudier les comportements des différents composants et leurs interactions en présence de fautes, au cours de l'utilisation du système par des applications clientes. Pour cela, une analyse approfondie de la structure fonctionnelle de chaque composant d'un système RFID a été menée dans le but d'en déterminer les fonctions principales. Les normes EPCglobal Class 1 Gen 2 (UHF) et ISO 18000-3 (HF) ont permis d'extraire les caractéristiques fonctionnelles des différents composants d'un système RFID et de leurs sous-composants [74].

Les figures 2.5 et 2.6 représentent respectivement le modèle fonctionnel proposé pour un tag ainsi que celui d'un lecteur.

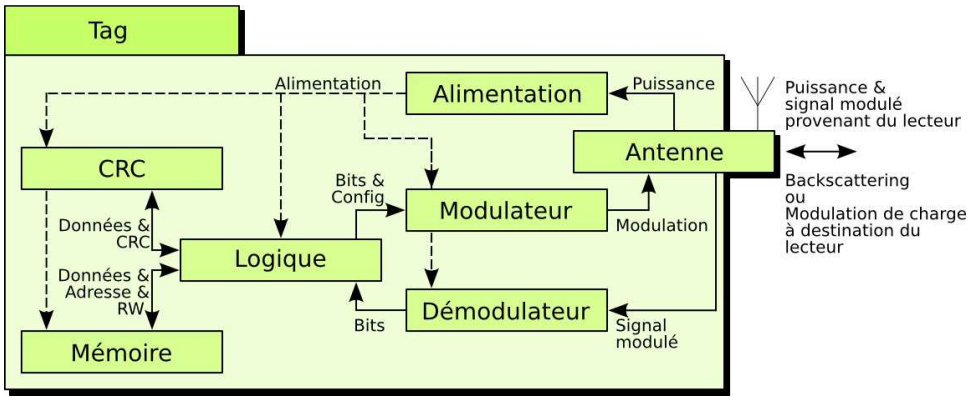

Figure 2.5. Modélisation du tag [74]

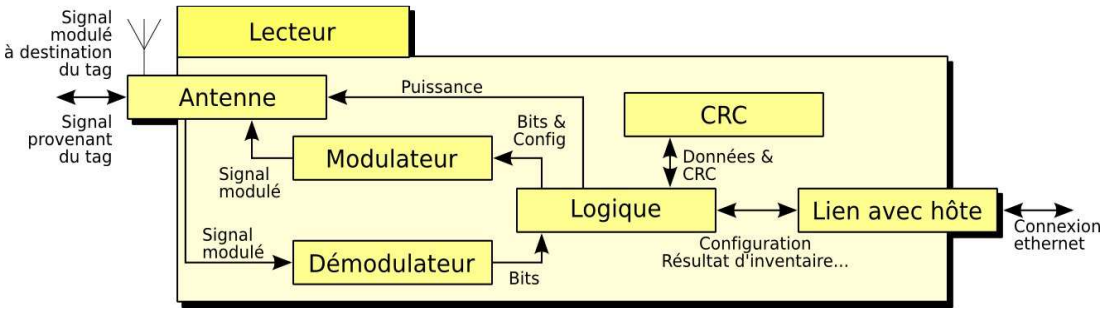

Figure 2.6. Modélisation du lecteur [74]

Ces figures montrent que les différents sous-composants identifiés ont été modélisés : antenne, modulateur, démodulateur, *etc*. en prenant bien en compte les caractéristiques et les phénomènes physiques tant pour modéliser le fonctionnement UHF que le fonctionnement HF. Les détails de cette modélisation sont présentés dans la thèse de G. Fritz [74] et ont été publiés dans [78][11]. Notons toutefois que les interactions entre les différents sous-composants et composants doivent pouvoir être observées. Le modèle est alors dit TLM, pour « *Transaction Level Model* » ou modèle au niveau transaction. La gestion du temps, importante pour l'évaluation du simulateur, est distribuée au niveau de chaque composant : le modèle est alors dit TLM-DT pour « *Transaction Level Model with Distributed Time* » ou modèle au niveau transaction avec temps distribué.

Tout comme cela est fait dans les simulateurs globaux existants, la propagation du signal analogique dans le canal de transmission ainsi que la modélisation des données binaires et du protocole de communication entre tags et lecteurs ont été pris en compte simultanément dans nos travaux grâce à l'utilisation de la bibliothèque SystemC.

[11] L'article correspondant est joint en Annexe B.

De nouveaux types de données ont ainsi été définis pour représenter d'une part les données numériques et d'autres part les signaux analogiques correspondants et permettant leur transmission entre le tag et le lecteur. Il est à noter que les signaux sont représentés ici par une information discrète qui comprend les principaux paramètres qui les caractérisent. Dès qu'un de ces paramètres change, le signal est mis à jour en conséquence. Cette solution évite l'introduction de paramètres physiques relativement complexes et longs à simuler, tout en mettant en avant ceux qui nous intéressent directement.

A titre d'illustration, le modèle d'un canal représentant le lien tag-lecteur inclut notamment comme caractéristiques la distance entre le tag et le lecteur, le taux d'erreurs binaires et la qualité[12] du signal. Il est donc possible par exemple de modifier la puissance de l'information transmise en fonction de la distance tag-lecteur. Pour cela, un facteur d'atténuation est implanté dans le canal. Pour le taux d'erreur binaire, la loi implantée est *la loi uniforme* qui permet de prendre en compte des erreurs de transmission tout en gardant des temps de simulation raisonnables.

Enfin, un dernier élément intervenant dans la communication tag-lecteur est représenté par l'environnement global dont le but est de prendre en compte les paramètres globaux communs à tous les équipements comme le bruit par exemple (voir figure 2.7).

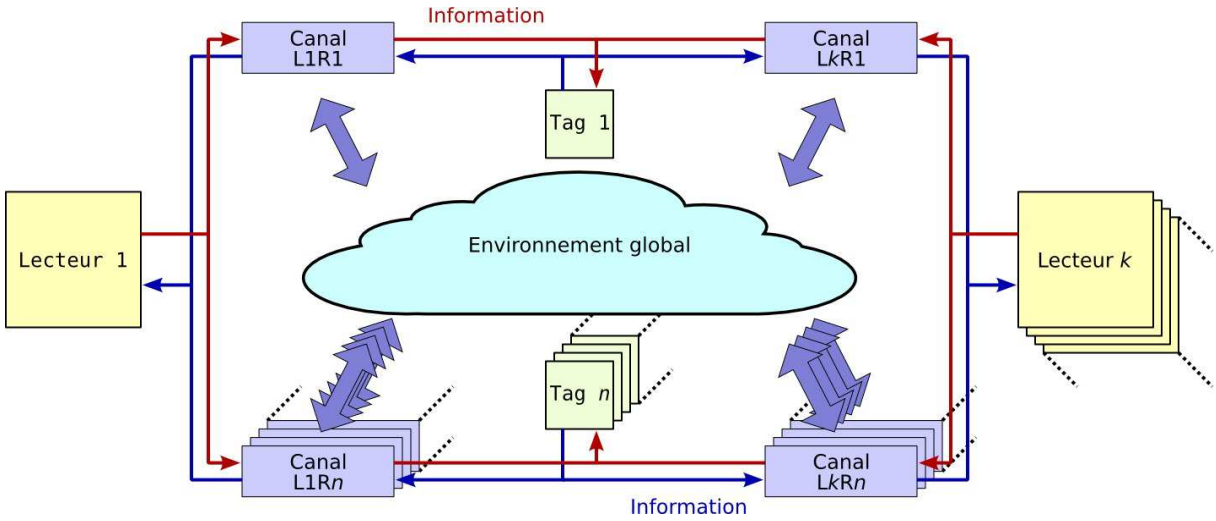

Figure 2.7. Interconnexion des différents modèles au sein du simulateur SERFID [74]

La figure 2.7 montre que le simulateur SERFID permet de prendre en compte les éléments suivants :

- simulation de plusieurs lecteurs et tags avec la spécification de chaque lien lecteur-tag existant.

[12] La qualité d'un signal permet de représenter divers paramètres physiques (rapport signal sur bruit, qualité de la synchronisation, distorsion du signal, *etc*.). Lorsque la qualité d'un signal est trop faible, il peut être impossible de démoduler et de décoder les informations qu'il contient.

- représentation des paramètres environnementaux communs à tous les équipements – l'environnement global.

Une validation *fonctionnelle* et *temporelle* de ce simulateur a permis de montrer que le simulateur donnait un comportement réaliste du système étudié. Cette validation s'est principalement focalisée sur l'analyse de la communication entre un lecteur et les tags lors de l'opération d'inventaire, c'est-à-dire la reconnaissance par le lecteur de l'ensemble des tags qui passent dans son champ. Les détails de cette validation sont présentés dans la thèse de Gilles Fritz [74] et ont été publiés dans [78].

La suite de ce chapitre présente l'approche de *test statistique Profil* que nous avons développée. Le simulateur SERFID sera alors utilisé pour analyser cette approche de test en permettant notamment l'injection de fautes dans le système étudié.

IV. Test en ligne et injection de fautes

IV.1. Les approches existantes

L'une des difficultés du test en ligne en général, et dans le cas des systèmes RFID en particulier, est de se dérouler dans un environnement non maîtrisable et non contrôlable et parfois même hostile. Les approches de test relèvent alors davantage de la supervision et de la surveillance du système et permettent de détecter des comportements fautifs à partir de l'observation du système. Ainsi, ces approches comprennent une étape de collecte d'informations liées aux performances du système suivie d'une étape d'analyse de ces informations en vue de déterminer une éventuelle déviation par rapport au comportement attendu.

Une classification classique des approches de test en ligne dans les systèmes RFID permet d'en distinguer deux familles : les *approches intrusives* qui nécessitent une modification du système et les *approches non-intrusives* qui se fondent sur l'analyse d'informations déjà disponibles au niveau du système, sans procéder à un changement de celui-ci [103] [104].

Les approches intrusives entraînent en général un recours à des informations autres que celles strictement nécessaires au fonctionnement des systèmes RFID. C'est le cas par exemple dans le contrôle de l'état d'un lecteur par un middleware RFID en utilisant des requêtes SNMP[13] pour vérifier que le lecteur est alimenté ou connecté ou encore que ses antennes sont opérationnelles, *etc*.

Dans les approches non intrusives, le caractère aléatoire du fonctionnement des tags et des lecteurs est pris en compte par la détermination d'un paramètre de performance caractéristique[14] et la surveillance régulière (continue ou périodique) de ses valeurs. On retrouve dans cette famille les approches fondées sur la surveillance du volume de tags passant devant un lecteur par unité de temps (ATTV ou Average Tag Traffic Volume) [103] [104] et les approches se basant sur la surveillance du nombre d'erreurs de lecture (RETR,

[13] SNMP (Simple Network Management Protocol) est un protocole de communication qui permet la gestion des équipements réseaux et le diagnostic de problèmes à distance.
[14] La détermination de la valeur appropriée du paramètre de performance choisi nécessite le plus souvent le recours à des techniques d'apprentissage.

Read Errors to Total Reads) [103] [104]. Une variation inattendue du paramètre choisi signale alors la présence d'une faute, sans pour autant la diagnostiquer.

L'approche ATTV permet de surveiller le nombre moyen de tags qui sont identifiés par un lecteur par unité de temps. Par exemple, si un lecteur identifie une moyenne de 100 tags par heure et qu'un jour, ce taux descend à 20 tags/h, alors cela peut être une indication sur une éventuelle défaillance du lecteur.

L'approche RETR analyse le nombre de tentatives de lectures échouées sur l'ensemble des tentatives du lecteur pour identifier tous les tags présents dans son champ de lecture (ce paramètre peut représenter plusieurs inventaires de tags). En effet, ce paramètre caractérise le fonctionnement des systèmes RFID [105] [106] **:** bien que la précision de ces systèmes RFID soit croissante, leurs taux de lectures durant un inventaire simple sont en moyenne de 60 à 70%. Il y a donc en général 30% d'erreurs de lecture : 30% des tags ne sont pas identifiés au cours d'un inventaire simple. Chaque application peut être caractérisée par un RETR dont les variations inhabituelles peuvent renseigner sur un dysfonctionnement du système.

IV.2. L'approche de test statistique par étude du profil de lecture

Lors des expérimentations effectuées conjointement avec le RFTLab, l'observation empirique des taux de lectures des tags a montré un comportement relativement « déterministe » de ces taux de lecture. Une analyse plus fine des valeurs obtenues nous a permis de proposer une approche de test statistique non intrusive, décrite ci-dessous.

Le développement de notre approche de test des systèmes RFID s'est ainsi déroulé en deux phases principales.

Phase 1 : observations empiriques

Dans un premier temps, nous avons procédé à des observations menées lors d'expérimentations répétées avec des inventaires applicatifs différents *in situ* sur un lecteur RFID et un ensemble de 110 tags, en utilisant les équipements du RFTLab[15] (voir figure 2.8). Ces observations ont révélé un comportement du taux de lecture très hétérogène d'un tag à un autre. Cependant, les nombres de lectures ordonnés (sans tenir compte de l'identité des tags) suivent la même forme, représentée sur la figure 2.9. Nous avons appelé cette forme le « *profil de lecture* » d'un ensemble de tags en fonction d'un lecteur donné. L'analyse de ces courbes nous a permis alors de proposer une nouvelle méthode de surveillance et de test en ligne des tags.

[15] Nous avons utilisé le lecteur ALR-8800 de la société *AlienTechnology* disponible au RFTLab. Ce lecteur UHF, respectant la norme EPCglobal Class 1 Gen 2, est un des plus utilisés dans l'industrie. L'objet de l'étude est une palette comportant 110 cartons de 6 bouteilles métalliques contenant du liquide, sur lesquels sont fixés 110 tags.

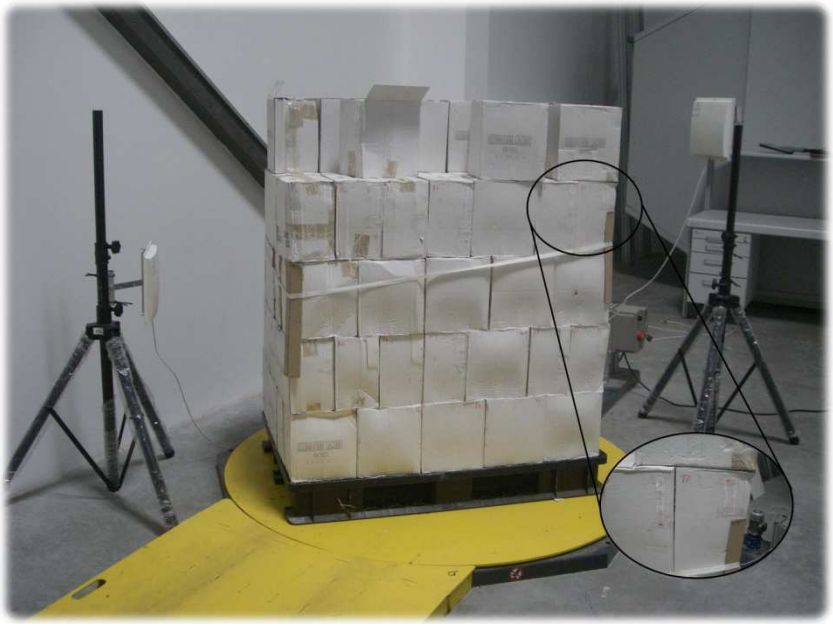

Figure 2.8. Représentation du système étudié [74]

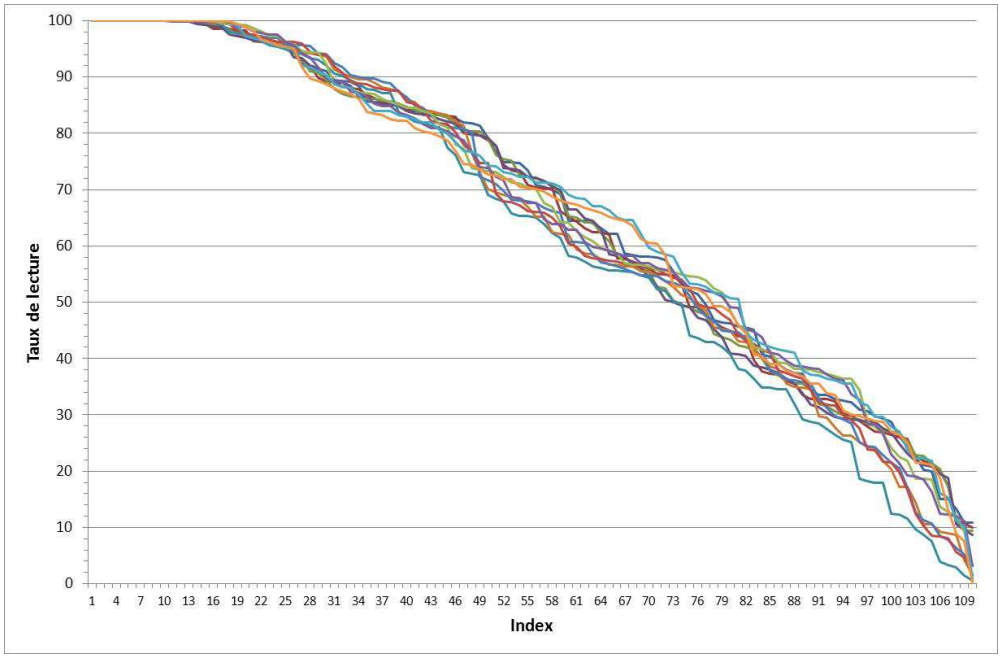

Figure 2.9. Taux de lectures ordonnés des tags de la palette [74]

Phase 2 : analyse théorique

L'observation des courbes obtenues nous permet de voir que, malgré l'absence de défaillances, certains tags peuvent ne pas être détectés. Ceci est dû à la configuration du système utilisé dans lequel certains tags sont enfouis et rencontrent alors de nombreux obstacles pour être lus. Le recours à des méthodes statistiques [107] nous a permis de déduire la distribution que suivent les nombres de lectures des tags. Dans le cas des expérimentations

effectuées[16], la distribution du nombre de lectures des tags suit la *loi normale*. En exploitant les propriétés gaussiennes de la loi normale, un « *profil limite* » a été déterminé. Il s'agit du profil qui se trouve au-dessous de 99,7 % des profils obtenus. En effet, les propriétés de la loi normale permettent de définir l'intervalle dans lequel se trouvent 99,7 % d'une population donnée en déterminant l'intervalle $[\bar{n} - 3.\sigma; \bar{n} + 3.\sigma]$ où $\bar{n}$ est la moyenne et σ l'écart-type de la distribution observée. Le profil limite de notre approche de test a été défini comme étant la borne inférieure de cet intervalle caractéristique. Comme les modèles de fautes étudiés se traduisent par une diminution du taux de lecture d'un tag, les profils des systèmes fautifs auront quelques points en-dessous du profil limite, ce qui permettra de détecter la présence de fautes (voir figure 2.10). Les détails de cette analyse ont fait l'objet des publications suivantes RFID-TA'11 [77], IMS3TW'10 [108] et DATE'12 [109][17].

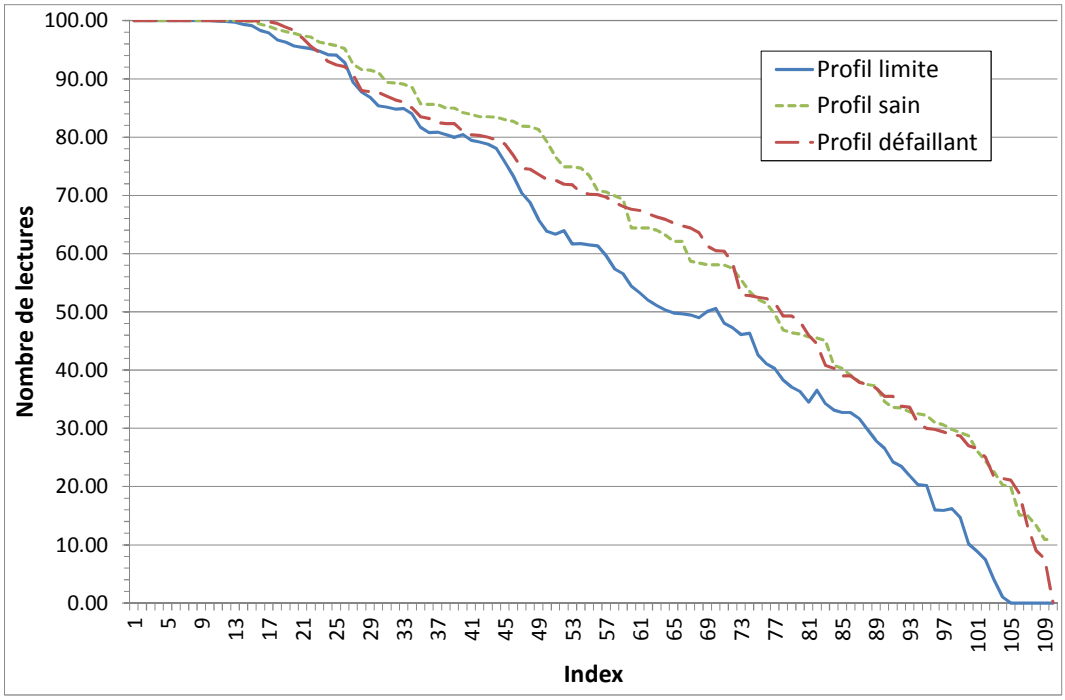

Figure 2.10. Comparaison entre le profil limite, un profil sain et un profil défaillant [74]

En conclusion, les observations empiriques que nous avons menées et l'analyse théorique simple qui s'en est suivie nous ont permis de proposer une nouvelle approche de test en ligne de tags en cours de lecture par un lecteur RFID.

IV.3. Evaluation de la méthode Profil

L'approche de test par l'étude du profil limite a été évaluée et comparée aux approches classiques ATTV et RETR de manière expérimentale sur le système du RFTlab décrit ci-dessus et par simulation en utilisant les modèles proposés et le simulateur SERFID développé. La simulation a permis une évaluation plus approfondie, puisque celle-ci est difficilement

[16] Nous avons opté dans ces expérimentations pour des inventaires applicatifs comportant chacun 100 inventaires simples, ce qui nous permet de déterminer à chaque fois le taux de lecture des différents tags.
[17] L'article publié dans DATE'12 est joint en Annexe B.

réalisable en pratique, notamment par l'injection de plusieurs fautes différentes. Il en ressort des performances de détection meilleures dans le cas de la méthode Profil lorsque l'environnement est homogène (lecture de palettes identiques). Cependant, l'utilisation conjointe des trois approches de détection RETR, ATTV et Profil améliore grandement le taux de détection, en raison des caractéristiques différentes de ces méthodes.

Les détails de ces évaluations sont présentés dans la thèse de G. Fritz [74] et ont fait l'objet des publications suivantes [77] [108] [109] [110].

Toutes ces approches de test et de surveillance des systèmes RFID ont pour point commun de ne s'appliquer qu'à un seul lecteur, ce qui dans le contexte des systèmes RFID actuels, qui mettent en œuvre plusieurs lecteurs en réseau, est insuffisant. En effet, en cas de problème, il est difficile d'en connaître l'origine exacte, notamment de distinguer une faute de lecteur et de tag. Par ailleurs, en considérant une approche qui tient compte de la présence de plusieurs lecteurs dans le système, la qualité et la précision du diagnostic peuvent être améliorées. Il s'agit du second volet de nos travaux sur les systèmes RFID dont les principaux aspects et résultats sont décrits dans les sections qui suivent.

V. Intégration du test et du diagnostic au niveau de l'intergiciel RFID

Le principe de l'approche que nous proposons consiste à analyser l'ensemble des résultats de lecture de plusieurs lecteurs RFID utilisés dans une application donnée, et d'en déduire un diagnostic précis sur la cause du comportement fautif observé. La réparation de l'élément fautif ou l'inhibition de son fonctionnement sont alors possibles. Une telle approche correspond tout à fait à l'évolution actuelle des systèmes RFID et à leur utilisation accrue dans des environnements distribués.

Le comportement aléatoire des systèmes RFID, en raison notamment de leur sensibilité à leur environnement, a orienté nos recherches vers des approches de *diagnostic probabiliste* [4] [111] [112] [113]. En effet, le diagnostic probabiliste permet de modéliser les phénomènes aléatoires en lien avec les dysfonctionnements observés dans un système. Il permet par exemple, de prendre en compte les *fautes permanentes* et les *fautes intermittentes*. Celles-ci sont souvent présentes dans ce type de systèmes au comportement aléatoire et dépendant de l'environnement. Pour cet aspect, notre travail s'inspire en particulier du travail de Fussel et Rangarajan sur le diagnostic probabiliste dans les systèmes distribués [111] [112].

Le travail effectué dans cet axe couvre les deux couches logicielles principales d'un système RFID c'est-à-dire la couche middleware ainsi que son interface de communication avec les sources de données, le protocole LLRP (Low Level Reader Protocol). Nous avons proposé une solution middleware compatible avec le standard LLRP, et utilisée comme un réceptacle pour une solution algorithmique de diagnostic probabiliste qui permet de détecter les défaillances potentielles des composants du système. Afin que le diagnostic soit le plus précis possible, un mécanisme d'analyse des fichiers de logs de l'interface de communication LLRP, complémentaire de l'algorithme probabiliste a été proposé. Il permet en particulier d'approfondir le diagnostic en recherchant les causes de la défaillance détectée. Enfin, nous

avons proposé une extension du standard de communication LLRP qui tient compte de plusieurs comportements défaillants dans le but de rendre ce dernier plus robuste. Ce travail de recherche a été au cœur de la thèse de R. Kheddam [75].

Dans le but de positionner notre travail par rapport à l'existant, nous passons tout d'abord en revue dans les paragraphes qui suivent, les principaux middlewares RFID connus, en mentionnant, lorsque cela est possible, la mise en œuvre de la sûreté de fonctionnement.

V.1. Existant en sûreté de fonctionnement et tolérance aux fautes dans les intergiciels RFID

Pour répondre aux demandes et aux besoins croissants des utilisateurs de systèmes RFID, plusieurs middlewares RFID ont été développés ces dernières années. Certains sont libres de droits alors que d'autres sont payants ou propriétaires. Les principales contraintes identifiées pour le développement de middlewares RFID sont : encapsulation des détails de la communication avec les lecteurs, gestion à large échelle du réseau, traitement et routage intelligents des données, interopérabilité entre le matériel et le logiciel, intégration et extensibilité du système [92].

Le développement d'un middleware RFID doit tenir compte de l'évolution technologique des standards et permettre aux applications, dans le meilleur des cas, d'être indépendantes de ces considérations ou de s'y adapter simplement.

Lorsqu'une solution de sûreté de fonctionnement existe dans ces middlewares, il s'agit souvent de la surveillance de base des lecteurs, comme mentionné précédemment à travers les méthodes RETR et ATTV notamment.

V.1.1. Middlewares RFID utilisant la surveillance de base des lecteurs

Comme indiqué en introduction, la sûreté de fonctionnement des systèmes RFID a très peu été étudiée contrairement à leur sécurité. La norme EPCglobal fournit toutefois des recommandations [114] allant dans le sens d'une meilleure robustesse pour les systèmes RFID. Ainsi, la plupart des travaux sur les middlewares RFID ont inclus une solution minimale de sûreté de fonctionnement reposant principalement sur la surveillance des lecteurs. Nous présentons ci-dessous quelques exemples de ces middlewares.

Fosstrak [115] [116]

Ce middleware, anciennement appelé Accada, a été développé sur la base des spécifications de la norme EPCglobal. Il se présente sous la forme de trois modules :

- Le module « lecteur » (Reader module) : son rôle est de gérer les lecteurs, de filtrer les données RFID et d'assurer leur transmission. Ce module est compatible avec les lecteurs de divers fournisseurs. Ce module informe le module "filtrage et collection » de la capture de nouvelles données.
- Le module « filtrage et collection » (filtering and collection middleware) : est notifié par le précédent composant quand de nouvelles données sont disponibles. Ce composant traite alors les données reçues selon les demandes formulées par les applications et les transmet au composant suivant.

- Le module « service d'informations EPC» (EPC Information services) : ce composant a pour charge de traduire les données RFID brutes en données et événements métiers avant de les transmettre aux applications finales.

De par le respect de la norme EPCglobal, le middleware Fosstrak réalise la surveillance des lecteurs, ainsi qu'un accès distant au lecteur lors de la maintenance. Cependant, il n'y a pas de mécanisme de tolérance aux fautes plus approfondi. Le middleware Fosstrak a été le noyau de base du projet AspireRFID décrit ci-après.

AspireRFID [117] [118]

AspireRFID (Advanced Sensors and lightweight Programmable middleware for Innovative RFID Enterprise applications) est un projet open-source européen, lancé dans la seconde moitié de l'année 2008 par le consortium OW2[18] pour le développement d'un outil open-source fiable dont le rôle est de faciliter le déploiement et la gestion des applications RFID.

Son architecture implémente plusieurs spécifications liées à la radio-identification comme celles de EPCglobal et NFC Forum. Ses composants principaux sont les suivants :

- La couche d'abstraction matérielle (Hardware Abstraction Layer, HAL)
  C'est la couche de plus bas-niveau. Elle fournit une abstraction du matériel afin d'unifier l'interaction du middleware avec les lecteurs de différents constructeurs, utilisant des protocoles de communication variés.
- Le mandataire de lecteur (Reader Core Proxy, RCP)
  Ce composant permet d'adapter les lecteurs non conformes au protocole utilisé par le middleware AspireRFID.
- Le filtrage et la collection des données (Filtering and Collecting, ALE)
  Cette couche fournit une interface flexible pour la spécification ALE[19] (Application Level Events) pour le filtrage des données produites en réponse aux requêtes des applications. Elle assure aussi la collecte et la fusion de données en provenance de plusieurs sources en vue de les transmettre aux applications clientes.
- Le générateur d'événements métiers (Business Event Generator, BEG)
  Ce composant se situe entre la couche de filtrage et le module de partage d'informations (EPCIS pour EPC Information Services). Son rôle est d'automatiser la correspondance entre les rapports produits par le composant de filtrage et les événements métiers du système.
- Integrated Development Environment (AspireRFID IDE)
  Il rend possible la gestion visuelle de tous les fichiers de configuration et des métadonnées qui sont requises pour le fonctionnement du système.

Le projet AspireRFID s'est essentiellement focalisé sur la réduction du coût de déploiement des applications et leur extensibilité.

[18] Ow2 est une communauté open-source créée en janvier 2007, et dédiée au développement des technologies middlewares libres et fiables [http://www.ow2.org]

[19] ALE est une spécification logicielle des fonctions d'un système RFID (notamment les opérations de lecture et d'écriture des tags) et toutes les fonctionnalités et le comportement qu'elles impliquent (comme le traitement des données).

Sun RFID [119] [120]

Le middleware SunRFID développé par l'entreprise Sun Microsystems (actuellement Oracle) prend en compte différents standards de la RFID dont EPCglobal. Sa conception a pour but de faciliter son intégration avec les applications existantes dans les entreprises. Son architecture comprend deux types de composants principaux :

- Le gestionnaire d'événements (Event Manager) : son rôle est de gérer les quantités de données brutes en provenance des lecteurs RFID. Ce composant assure la capture, le filtrage et la communication des données RFID au serveur d'informations ainsi qu'aux applications clientes.
- Le serveur d'informations (Information Server) : il s'agit d'une application J2EE[20] qui sert d'interface à la capture et la recherche de données RFID. Ces données sont agrégées et stockées à ce niveau. Elles sont également présentées aux applications après leur transformation d'un bas niveau d'observation vers un haut niveau de représentation.

A notre connaissance, les aspects liés à la sûreté de fonctionnement ne sont pas traités dans ce middleware.

WinRFID [121] [122] [123]

Ce middleware a été développé dans le cadre du consortium WinMec [124] à l'Université de Californie à Los Angeles. Tout comme le middleware SunRFID, le principal objectif à travers le développement de ce middleware est de faciliter l'intégration de la RFID dans des applications existantes. Ce middleware réalise les fonctions attendues d'un middleware RFID sous la forme d'une architecture à 5 couches :

- Une couche matérielle, responsable de l'interfaçage avec les dispositifs matériels (lecteurs, tags, capteurs, *etc*.) Elle permet de tenir compte de dispositifs en provenance de divers vendeurs et fonctionnant selon plusieurs standards RFID (LF, HF ou UHF)
- Une couche protocolaire qui permet le support des multiples protocoles de communication (ISO 15693, ISO 18000-6B, ICode, EPC Class 0, *etc*.) et qui transmet dans un sens les requêtes émanant des couches supérieures et dans l'autre sens les données en provenance des tags.
- Une couche gestion de données ; c'est la couche où se réalisent l'agrégation et le filtrage des données en provenance des tags.
- Un framework XML qui assure le formatage des données reçues en utilisant le standard XML pour une meilleure interopérabilité entre les applications.
- Une couche présentation des données qui permet aux utilisateurs de spécifier les informations qui leur sont utiles et de les récupérer de manière sécurisée et lisible.

De par sa gestion sûre des données et la surveillance des lecteurs, le middleware WinRFID présente un certain degré de tolérance aux fautes. Toutefois, ce sont principalement les mécanismes de sécurité qui y sont mis en œuvre de manière approfondie.

[20] J2EEest une spécification de la technologie Java pour les applications d'entreprise. Il s'agit d'une extension de la technologie Java standard en vue de permettre le développement d'application multi-niveaux et réparties.

WebSphere [125]

Ce middleware RFID a été développé par la société IBM. Son principal apport comparativement aux autres middlewares a été de déporter le traitement des données sur les lecteurs qui sont des équipements de plus en plus intelligents. Cette solution permet également d'intégrer les applications RFID aux applications existantes, enrichissant ainsi le traitement des données au niveau des entreprises.

Le middleware WebSphere réalise une partie de la norme EPCglobal et se base sur les technologies J2EE et OSGi[21]. Il comporte trois composants principaux :

- Premises Server (PS) : est une application J2EE assurant la communication entre le monde physique (lecteurs RFID) et le monde des technologies de l'information liées au métier. Ce composant permet en particulier d'optimiser la solution et de la personnaliser en fonction des demandes de chaque entreprise utilisatrice.
- Device Infrastructre (DI) : c'est une plateforme développée avec OSGi qui fournit aux différents fabricants de lecteurs intelligents des technologies avancées (sous licence) pour l'exploitation des données RFID au niveau des lecteurs.
- Business Integrated Server (BIS) : est le composant qui assure la connexion de la solution WebSphere au système d'informations de l'entreprise.

De même que pour les middlewares précédents, le recours à la norme EPCglobal permet la prise en compte de mécanismes de base pour la surveillance des lecteurs, mais aucune solution de tolérance aux fautes n'est réellement développée.

REFiLL [126]

REFiLL est un middleware RFID basé sur le standard EPCglobal. Sa particularité est de présenter un environnement programmable pour permettre aux développeurs des applications RFID de s'adapter à des plateformes matérielles (lecteurs) différentes. Deux composants permettent le fonctionnement du middleware REFiLL :

- Les filtres : construits à l'image des filtres classiques du système Unix, et décrits dans des fichiers XML. Ces filtres sont ensuite compilés en code Java pour fournir les fonctions de filtrage et de collection des données RFID.
- Les output managers : reçoivent les données des filtres et permettent de les présenter aux utilisateurs selon le format des données métier.

A notre connaissance, ce middleware ne comprend pas de solution spécifique pour la sûreté de fonctionnement.

V.1.2. Middlewares proposant une approche globale de tolérance aux fautes

Contrairement aux middlewares ci-dessus, le middleware RF^{2}ID (Reliable Framework for Radio Frequency Identification) [127] [128] intègre une solution de sûreté de fonctionnement

[21] OSGi (Open Service Gateway initiative) déjà évoqué dans le Chapitre 1, offre un framework de composants logiciels permettant la livraison de services administrés, dédié initialement aux passerelles de services mais dont l'utilisation s'est étendue notamment dans des réseaux résidentiels, les environnements restreints (voitures, *etc.*), les serveurs d'appliactions, *etc*. [23].

qui vise à améliorer la traçabilité des éléments lus par les applications. Ce middleware introduit la notion de *lecteur virtuel* (VR, *Virtual Reader*) qui regroupe un ensemble de *lecteurs physiques* (PR ou *Physical Reader*) selon le critère du voisinage géographique ainsi que la notion de *chemin virtuel* (VPath , *Virtual Path*) pour capturer le flux logique des données à travers les lecteurs virtuels [129]. Ainsi, un VPath est un canal logique entre plusieurs VR. Il est créé dynamiquement en fonction des paramètres qui caractérisent les lecteurs virtuels (VR) qu'il met en relation.

La liste des tags observés d'un lecteur virtuel VR est l'ensemble des tags observés par tous les lecteurs physiques appartenant au lecteur logique, en éliminant les redondances. Le cheminement des tags observés est vérifié entre deux lecteurs virtuels voisins VR_{i-1} et VR_i et toute différence entre les observations de tags effectuées par VR_i et VR_{i-1} permet à VR_i de déterminer les tags attendus mais qui ne sont pas vus, ou les tags non demandés mais qui sont vus. En outre, c'est en observant les informations qui transitent sur un chemin virtuel qu'il est possible d'effectuer les opérations suivantes :

- Localisation d'un objet vu pour la dernière fois par un VR ou un PR ;
- Information concernant un lecteur fautif (qui perd la trace des objets de manière régulière) ;
- Information sur une chaîne de transmission défaillante (un VPath qui perd des objets de manière régulière).

Le fonctionnement du middleware RF^{2}ID permet de remonter la liste des lecteurs virtuels jusqu'au dernier ayant vu le tag perdu. Il est ensuite possible de localiser le PR en charge de ce tag et de le configurer (notamment au niveau de sa puissance de fonctionnement) de sorte à rendre possible la lecture du tag non vu précédemment.

L'exemple de ce middleware montre que l'élargissement, au-delà d'un lecteur unique, des informations prises en compte pour l'analyse du fonctionnement d'un système RFID, permet d'observer et de traiter davantage de dysfonctionnements ; offrant ainsi aux applications un service de sûreté de fonctionnement riche et une robustesse renforcée. C'est ce même principe d'une décision globale au niveau du système qui est adopté dans notre travail. L'apport principal de notre contribution consiste à prendre en compte le comportement aléatoire inhérent aux systèmes RFID.

En conclusion, il est possible de noter que peu de middlewares RFID existants intègrent une solution de sûreté de fonctionnement élaborée. C'est le cas à notre connaissance du middleware RF^{2}ID. La plupart des autres middlewares, lorsqu'ils prennent en compte les aspects liés à la robustesse des systèmes RFID, le font à travers des approches de monitoring. Or, ces approches se limitent à indiquer la présence de fautes, sans toutefois permettre leur diagnostic. De plus, la solution proposée par le middleware RF^{2}ID se limite à considérer l'inventaire des tags selon leur visibilité par les lecteurs uniquement. Même si elle est bien adaptée pour la surveillance des mouvements des tags, elle ne permet pas de détecter les erreurs de lecture ou les baisses de performances d'un lecteur. C'est pourquoi, nous nous intéressons dans la suite de ce travail au développement d'une approche de diagnostic globale,

qui se fonde sur une méthode de test basée sur les performances de lecture des différents lecteurs, en y incluant une analyse probabiliste globale.

V.2. Etude AMDE et modèles de fautes

Tout comme pour les composants matériels (lecteurs et tags), l'étude des parties logicielles d'un système RFID a nécessité le recours à une AMDE pour en appréhender les principaux aspects. Plus précisément, deux études AMDE ont été développées :

- la première étude a concerné le middleware RFID tel que spécifié dans la norme EPCglobal [130].
- la seconde étude a concerné le protocole LLRP (Low Level Reader Protocol) qui est le standard de communication entre le middleware et les lecteurs dans la norme EPCglobal [80].

L' AMDE du middleware RFID a permis d'analyser de manière approfondie les trois couches principales : la couche d'interface avec les applications (AAL : Application Abstraction Layer), la couche de traitement des données (DPL : Data Processing Layer) et la couche d'abstraction matérielle (HAL : Hardware Abstraction Layer). Cette étude est détaillée dans la thèse de R. Kheddam [75].

Les défaillances auxquelles nous nous intéressons peuvent être classées en trois catégories :

- L'apparition ou la disparition d'un tag,
- L'absence de réponse d'un lecteur
- La dégradation des performances d'un lecteur (perturbations externes, détérioration d'une antenne, *etc.*)

Ces défaillances peuvent être dues à des causes variées, dont les principales que nous avons identifiées sont :

- Des erreurs de conception ou de programmation des différents composants d'un middleware RFID ainsi qu'une mauvaise configuration des paramètres du middleware ou du protocole,
- Des défauts au niveau des composants matériels,
- Des erreurs liées aux conditions d'exécution, notamment la surcharge du système,
- Des erreurs au niveau des interactions entre les différents composants internes du middleware.

Par la suite, nous avons affiné le choix des défaillances auxquelles nous nous intéressons en vue du diagnostic en en identifiant principalement deux catégories :

- Les défaillances dues à des fautes de conception ou de configuration,
- Les défaillances dues à des fautes en lien avec l'environnement d'exécution.

Les défaillances dues à des fautes de conception ou de configuration considérées sont :

- Masquage ou blocage des données par le middleware,
- Incohérence entre les données reçues et les données attendues de la part du lecteur ou du middleware.

Les défaillances dues à des fautes en lien avec l'environnement d'exécution prises en compte sont :

- Exécution lente, surcharge,
- Pas de capture de données,
- Données reçues erronées.

Le modèle de fautes auquel nous nous intéressons couvre ainsi des fautes temporaires et des fautes permanentes et représente des degrés variés de sévérité des dysfonctionnements : composant (tag ou lecteur) hors service, composant dégradant les performances du système (fréquence d'erreurs plus élevée, vitesse de lecture plus faible, *etc*.)

L'AMDE basée sur la description fonctionnelle du protocole LLRP a permis de relier, lorsque cela se justifie, les principales défaillances identifiées au niveau du middleware à des causes internes au protocole LLRP. Par exemple, l'absence d'identification d'un tag peut être due à de mauvais réglages au niveau des commandes LLRP. Cet aspect de notre travail a notamment permis la mise en place d'une analyse automatique des fichiers de logs du protocole LLRP.

V.3. Diagnostic probabiliste dans les systèmes RFID

Les approches de monitoring existantes (ATTV, RETR, *etc*.) ainsi que celle basée sur la méthode Profil proposée dans nos travaux, signalent l'occurrence d'une faute au niveau d'un lecteur unique par l'observation des variations d'un paramètre de performances. Il n'est alors pas aisé de déterminer l'origine de cette erreur : interne au lecteur même, située au niveau du groupe de tags lus par le lecteur, due à une perturbation au niveau de l'environnement du système, ou à une combinaison de quelques-unes ou de l'ensemble de ces causes possibles.

En positionnant cette partie de notre étude des systèmes RFID au niveau du middleware, l'observation simultanée de plusieurs lecteurs lors de la lecture d'un ou de plusieurs groupes de tags est rendue possible. L'analyse des résultats produits par ces observations permet d'affiner le diagnostic et de localiser plus précisément l'origine probable d'une défaillance. Nous avons opté pour une comparaison des résultats d'inventaires de plusieurs lecteurs comme cela est fait dans le middleware RF^{2}ID. Toutefois, dans notre approche, les comparaisons sont opérées sur des lecteurs physiques qui lisent les mêmes groupes de tags et tiennent compte des réelles performances de fonctionnement de chaque lecteur.

La nature aléatoire du comportement du couple tag-lecteur a orienté nos recherches vers le diagnostic probabiliste [4] [112] [111] [113] dont le principe de base consiste à associer une probabilité de panne à chaque élément dans le système ainsi qu'un taux de couverture de fautes à chaque test réalisé. Le diagnostic indique alors un « niveau de confiance justifiée » que l'utilisateur peut avoir dans le résultat fourni.

Le processus de diagnostic que nous proposons se déroule en deux phases.

La première phase s'inspire du travail de Rangarajan et Fussel [112] [111] sur le diagnostic probabiliste dans les architectures distribuées ainsi que de l'approche de localisation des lecteurs défaillants proposée dans le middleware RF^{2}ID. La combinaison du principe de base de ces deux approches nous permet de proposer une approche réaliste du

point de vue opérationnel et mesurable du point de vue qualitatif grâce à l'utilisation de calculs probabilistes. En particulier, nous avons procédé à une analyse probabiliste des résultats obtenus par les lecteurs lors d'inventaires de groupes de tags en vue de déterminer les éléments fautifs.

Chaque diagnostic ainsi déduit est évalué par un paramètre de confiance que nous avons appelé *identifiabilité* et qui est décrit ci-après.

La deuxième phase s'exécute pour chaque lecteur identifié défaillant. Elle se base sur l'analyse de la communication entre le middleware et le lecteur pour cerner la cause précise de la défaillance observée, sachant qu'une telle information n'est pas disponible dans le protocole LLRP et constitue donc également un aspect novateur de nos travaux.

La suite de cette section décrit le déroulement de ces deux phases.

V.3.1. Description de la première phase : le diagnostic probabiliste global

Dans l'approche que nous proposons, les lecteurs RFID sont divisés en groupes selon le cheminement réel des tags à travers les différents lecteurs du système étudié. Chaque groupe de lecteurs issu de ce partitionnement comprend par conséquent des lecteurs qui traitent les mêmes ensembles ou groupes de tags[22]. Lorsqu'un certain nombre de groupes de tags ont été lus par un ensemble de lecteurs qui leur sont affectés, une analyse probabiliste des résultats de lecture est effectuée. A ce niveau, un résultat de lecture est déduit de l'application d'une des approches de monitoring des lecteurs vues précédemment. Cela peut être l'approche ATTV, RETR ou la méthode Profil. La nature du monitoring utilisé importe peu, puisque seule l'indication de la qualité de l'inventaire pour chaque lecteur et chaque ensemble de tags est prise en compte.

Tableau 2.1. Comparaison des résultats de lecture

	g_1	g_2	g_3
R_1	1	0	1
R_2	0	1	1
R_3	1	0	1
R_4	1	0	1
R_5	1	0	-
R_6	-	-	1
R_7	-	-	0

Il est possible d'observer sur le tableau 2.1. les différents résultats de lectures collectés pour un ensemble de lecteurs (R_1, R_2, …, R_7) et 3 groupes de tags g_1, g_2 et g_3. La valeur « 1 »

[22] Nous avons fait le choix de traiter les tags par groupes et non pas individuellement car le diagnostic proposé se base sur des informations liées aux performances de fonctionnement des lecteurs. Ces informations sont issues de techniques de monitoring qui traitent les tags par groupes. Dans la thèse de R. Kheddam, il est indiqué toutefois que les performances du diagnostic sont optimisées lorsque les tags sont considérés simultanément par groupes et individuellement.

indique que le lecteur a assuré un inventaire correct de l'ensemble de tags alors que la valeur « 0 » indique un échec de cet inventaire. Lorsqu'un lecteur n'est pas concerné par un groupe de tags, le croisement de la ligne et de la colonne correspondantes comprend le symbole « - ».

L'analyse des résultats obtenus se fait alors en considérant les variations du paramètre de performance choisi suivant chaque groupe de tags, en appliquant le principe du vote majoritaire dans les deux cas suivants [75] :

1. Si la majorité des lecteurs respectent le paramètre de performance défini, alors le reste des lecteurs (la minorité) sont considérés défaillants ; *i.e.*, la minorité qui montre des performances faibles par rapport à la normale est considérée défaillante.
2. Si la majorité des lecteurs ne respectent pas le paramètre de performance défini sur un groupe de tags, alors ce groupe de tags, ainsi que les autres lecteurs (la minorité) sont considérés défaillants.

Cette phase du diagnostic probabiliste se termine par la détermination d'un paramètre de précision qui correspond au niveau de confiance que l'utilisateur peut avoir dans les décisions du diagnostic. En particulier, il est nécessaire d'assurer les deux points suivants :

- Un lecteur valide doit être identifié comme valide (ce cas est appelé « *correct negatif* » et noté CN).
- Un lecteur défaillant doit être identifié comme défaillant. Pour simplifier les calculs probabilistes, nous considérons le cas complémentaire, c'est-à-dire le cas où un lecteur défaillant est considéré correct (ce cas est appelé « *false negatif* » et noté FN)

Nous appelons *identifiabilité* la probabilité d'identifier correctement l'état des lecteurs diagnostiqués. Cette mesure indique donc la capacité du diagnostic de « bien distinguer » les lecteurs défaillants de ceux qui ne le sont pas. Cette probabilité représente la précision du diagnostic et elle est donnée par la formule probabiliste suivante :

$$I(t, n, p, r) = (1 - p) \times CN(t, n, p, r) + p \times (1 - FN(t, n, p, r))$$

où :

- n est le nombre total des lecteurs,
- t est le nombre de groupes de tags,
- p est la probabilité initiale de défaillance d'un lecteur[23],
- r (*Rational Behavior)* est la probabilité que la défaillance d'un lecteur se manifeste[24],
- $CN(t, n, p, r)$ et $FN(t, n, p, r)$ sont respectivement la probabilité d'avoir un correct négatif et la probabilité d'avoir un faux négatif. Les spécifications détaillées de l'algorithme de diagnostic probabiliste ainsi que le détail des calculs de ces probabilités sont donnés dans [75] [79].

[23] Pour ne pas alourdir les différentes formules probabilistes, nous considérons que les lecteurs ont la même probabilité de défaillance. Le cas de probabailités différentes pour les différents lecteurs est traité dans la thèse de R. Kheddam.

[24] Des indications pour la déterminsation de ce paramètre sont données dans la thèse de R. Kheddam.

V.3.2. Description de la seconde phase : Diagnostic d'un lecteur défaillant

Une fois les lecteurs défaillants identifiés, une étude supplémentaire nous a permis de localiser précisément la cause de la défaillance observée. Pour cela nous avons eu recours à l'analyse de la communication entre le middleware et le lecteur RFID et à l'étude du protocole LLRP qui gère cette communication.

En effet, bien que ce soit un protocole de communication complet et complexe qui permet de notifier les erreurs de communication entre le middleware et les lecteurs, le protocole LLRP ne peut pas détecter des défaillances de lecteurs dues à une mauvaise configuration, ni déterminer pour un comportement fautif donné, la cause de ce comportement. De ce fait, il est peu adapté en l'état à l'utilisation dans des applications où les demandes de sûreté de fonctionnement sont importantes, d'autant plus que l'interface tag-lecteur est très sensible aux perturbations externes et présente ainsi un comportement très aléatoire. En outre, le fonctionnement du protocole LLRP est relativement souple et fournit une large autonomie à l'application pour spécifier les opérations d'inventaires et d'accès aux tags. Ceci peut entraîner des erreurs de configuration résultant en un comportement fautif des lecteurs (par exemple, le lecteur n'arrive pas à identifier tous les tags dans son champ, le lecteur ne retrouve pas les informations correctes concernant les tags, *etc.*).

Notre travail sur le protocole LLRP permet principalement de remédier à ces limitations.

L'étude du protocole LLRP a conduit à sa modélisation sous la forme d'une machine à états finis[25].

Notons *G* la machine à états finis du protocole LLRP. $G = (S, I, O, \delta, \lambda)$; où *I*, *O* et *S* sont respectivement un ensemble fini de symboles d'entrée, un ensemble fini de symboles de sortie et un ensemble fini d'états.

- $\delta : S \times I \rightarrow S$ est la fonction de transition d'états.
- $\lambda : S \times I \rightarrow O$ est la fonction de sortie.

Quand un lecteur RFID ou un middleware est dans un état *s* de *S*, et reçoit une entrée *a* de *I*, il produit une sortie spécifiée par $\lambda(s, a)$ et passe vers un état spécifié par $\delta(s, a)$.

Les étiquettes des transitions dans l'automate LLRP sont sous la forme « X/Y », où X représente les messages (généralement des requêtes) du middleware vers le lecteur, et Y représente généralement la réponse du lecteur au message X du middleware.

Nous rappelons ci-dessous les défaillances étudiées (*cf.* Chapitre 2, section V.2).

- Les défaillances dues à des fautes de conception ou de configuration :
  - Masquage ou blocage des données par le middleware
  - Incohérence entre les données reçues et les données attendues de la part du lecteur ou du middleware
- Les défaillances en lien avec l'environnement d'exécution :
  - Exécution lente, surcharge
  - Pas de capture de données

[25] La machine à états finis que nous proposons pour le protocole LLRP est détaillée dans la thèse de R. Kheddam [75]et son utilisation pour le diagnostic présentée dans l'article [130].

- Données reçues erronées

Pour le premier type de fautes (conception ou configuration), le comportement erroné est associé à un état incohérent au niveau du lecteur ou du middleware en lien avec les données ou les requêtes reçues. L'entité (lecteur ou middleware) qui se trouve dans un état incohérent ne peut pas interpréter correctement les données reçues et adopte un comportement inapproprié. Pour traiter ce type de fautes, nous avons appliqué sur la machine à états finis du protocole LLRP des techniques classiques de test à base de modèles. Il s'agit ici de l'approche des séquences distinctives (*distinguishing sequences*) [60]. Une séquence distinctive mène toujours vers un état final unique. De même, l'inverse d'une séquence distinctive doit aussi mener vers un état unique pour que la séquence distinctive d'un état soit valide. L'application de cette technique permet de retrouver simplement l'état dans lequel se trouvait le système au moment de l'occurrence de la défaillance [130][26] [131]. Cette technique présente cependant une limitation car tous les états de la machine à états ne possèdent pas forcément une séquence distinctive[27].

Le second groupe des défaillances identifiées ne peut pas être lié directement à un état du système, puisqu'il s'agit de défaillances dues à l'environnement d'exécution. Ainsi, il n'est pas possible d'appliquer simplement l'approche précédente. Nous avons donc proposé une extension de la machine à états de manière à inclure les causes de cette catégorie de défaillances dans le processus de diagnostic sous la forme d'un modèle LLRP étendu. Ainsi, à la détection d'un dysfonctionnement, le système migre vers un état du modèle étendu dans lequel le diagnostic est exécuté pour permettre la détermination de la cause de la défaillance observée. Le processus de diagnostic commence par vérifier les fautes les plus simples à diagnostiquer. Par exemple, dans le cas d'une communication défaillante, la vérification porte dans un premier temps sur l'état des antennes (des requêtes spécifiques permettent de demander l'état des antennes au lecteur concerné). Si cette vérification s'avère négative, le diagnostic procède à la vérification suivante qui consiste à vérifier si les tags sont en dehors du champ du lecteur, *etc*. Les extensions apportées au protocole LLRP permettent de déterminer avec précision la cause ou les causes possibles d'un comportement défaillant observé, ce qui à notre connaissance représente une nouvelle fonctionnalité dans le domaine des middlewares RFID [75] [79].

V.4. Evaluation

Divers aspects de ce travail ont pu être évalués. Tout d'abord, nous avons étudié le comportement du modèle probabiliste proposé et en particulier de la précision du diagnostic ou identifiabilité en fonction de divers paramètres (nombre de lecteurs, nombre de groupes de

[26] L'article correspondant est joint en Annexe B.

[27] Les séquences distinctives présentent d'autres inconvénients, notamment leur caractère intrusif et l'absence de garantie des résultats lorsqu'elles sont appliquées sur un système fautif qui se bloque par exemple. C'est pourquoi, une analyse de fichiers de logs liés au fonctionnement du protocole LLRP complète en pratique l'algorithme de diagnostic dans le cadre du projet SafeRFID [75] [81].

tags, paramètres p indiquant la probabilité qu'un lecteur soit défaillant, paramètre r indiquant la probabilité de manifestation du comportement erroné d'un lecteur, *etc*.)

Quelques résultats sont présentés dans la thèse de R. Kheddam. Un des résultats concerne la variation de l'identifiabilité en fonction du paramètre r (manifestation de la faute des lecteurs).

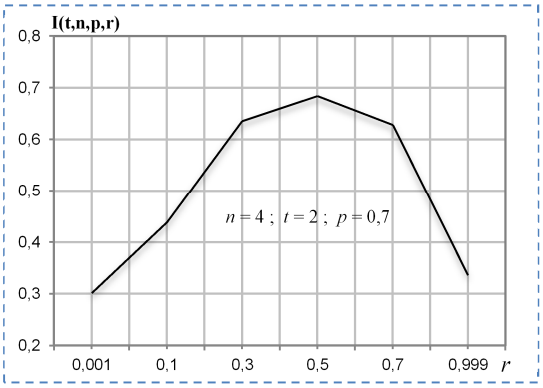

Figure 2.11. Variations de l'identifiabilité suivant le paramètre r

La figure 2.11 a été obtenue avec la valeur de p = 0.7, ce qui signifie que la plupart des lecteurs sont défaillants. Dans ce cas, la précision du diagnostic dépend fortement de la manifestation des fautes au niveau du comportement du système. Ainsi, les observations suivantes peuvent être faites :

- Quand r est petit, la plupart des lecteurs défaillants ne manifestent pas leur défaillance, et donc ils ne sont pas détectés vu qu'ils ont les mêmes résultats que les lecteurs corrects (*i.e.*, la majorité des lecteurs sont défaillants et ne manifestent pas leurs défaillances).
- Avec l'augmentation de r jusqu'à atteindre environ 0.5 ; (*i.e.*, le nombre de lecteurs qui manifestent leurs défaillances augmente), l'identifiabilité I augmente, vu que tous les lecteurs qui manifestent leurs défaillances représentent la minorité et sont donc détectés.
- Une fois que r a dépassé 0.5, son effet sur I s'inverse ; *i.e.*, la plupart des lecteurs défaillants manifestent leurs défaillances et donc, ils représentent la majorité et par conséquent ils sont considérés comme non défaillants (*i.e.*, l'identifiabilité diminue).

Un prototype du middleware SafeRFID_MW a également été développé et utilisé pour évaluer l'algorithme de diagnostic probabiliste proposé. L'algorithme de diagnostic en soi s'exécute assez rapidement (la complexité de l'algorithme, sans optimisation, est évaluée à où n et t sont respectivement le nombre total des lecteurs en cours d'analyse et le nombre total de groupes de tags). Toutefois, la phase de collecte des résultats des lecteurs prend davantage de temps et peut pénaliser la transmission rapide des résultats de diagnostic. Pour remédier à cela, diverses conditions de déclenchement du diagnostic ont été analysées, dont :

- Déclenchement du diagnostic à la fin de la collecte des résultats de tous les lecteurs,

- Déclenchement du diagnostic dès la disponibilité des résultats de cinq lecteurs, selon deux modes de fonctionnement : un regroupement cinq par cinq des lecteurs, ou l'utilisation d'une fenêtre glissante qui permet d'avoir à disposition les résultats de cinq lecteurs, en en remplaçant un à la fin de chaque processus de diagnostic

Les résultats obtenus concernant ces différentes implémentations sont présentés dans [75] [79][28].

V.5. Autres travaux

En parallèle aux travaux menés sur les systèmes RFID, le projet CEDRE mené en collaboration avec l'Université Libanaise (le laboratoire LASTRe) et qui a été le cadre de la thèse de Doctorat de Dima Hamdan [132], a permis l'exploration de la sûreté de fonctionnement d'une autre catégorie de systèmes combinant des composants matériels et d'autres logiciels dans un cadre embarqué et réparti. Il s'agit des systèmes à base de réseaux de capteurs. Les similitudes de ce type de systèmes avec les systèmes RFID ont permis d'approfondir la problématique de la sûreté de fonctionnement d'une catégorie importante d'applications embarquées qui connaît actuellement un essor grandissant du fait de l'Internet des Objets notamment.

Le travail de thèse de D. Hamdan a permis d'explorer différents aspects de la sûreté de fonctionnement des systèmes à base de réseaux de capteurs en abordant la problématique selon trois niveaux : le niveau des *données*, le niveau des *nœuds capteurs* et le niveau *applicatif*. Pour chaque niveau, une approche adaptée a été proposée en tenant compte de l'existant dans le domaine et des contraintes posées dans le cadre de nos travaux concernant la minimisation du coût de la sûreté de fonctionnement. Nous résumons ci-dessous les principaux résultats obtenus :

- Le développement d'une méthode d'analyse des données des capteurs selon une approche d'algorithmes génétiques pour détecter les données erronées [133].
- Une approche de monitoring des nœuds capteurs basée sur un protocole de surveillance mutuelle entre nœuds voisins [134].
- Le développement d'un service de test et de diagnostic de niveau applicatif, appelé IFTF (Integrated Fault-Tolerance Framework) qui intègre et exploite les approches précédentes [135].

V.6. Bilan et perspectives

V.6.1. Bilan

Ce chapitre a présenté l'ensemble des méthodes et des approches proposées pour améliorer la sûreté de fonctionnement des systèmes RFID. En particulier, les résultats suivants ont été obtenus :

- Une approche de *test statistique* des tags RFID dans le but de détecter les défauts de lectures. Cette approche, appelée méthode *Profil*, est une approche de test

[28] L'article correspondant est joint en Annexe B.

empirique fondée sur l'observation et l'analyse des taux de lectures des tags dans un système RFID.

- Le développement d'un simulateur appelé SERFID pour la *simulation des systèmes RFID HF et UHF*. Ce simulateur a notamment été utilisé pour valider l'approche de test Profil.
- La spécification et le développement d'un prototype de middleware RFID appelé SafeRFID-MW. Ce middleware est capable de diagnostiquer les lecteurs et les tags défaillants dans un système RFID selon une approche de *diagnostic probabiliste*. Ce diagnostic est complété par une analyse fine des fichiers de logs des échanges entre le middleware et les lecteurs RFID en vue de localiser précisément l'origine des dysfonctionnements observés.

Ces travaux ont eu pour cadre le projet ANR SafeRFID. Deux thèses de Doctorat ont été co-encadrées dans le cadre de ces travaux : la thèse de Gilles Fritz [74] sur le développement du simulateur SERFID et la méthode Profil et la thèse de Rafik Kheddam [75] sur le développement du middleware et de l'approche de diagnostic probabiliste.

Par souci de concision, seuls les résultats obtenus pour les systèmes RFID ont été détaillés dans ce chapitre. Cependant, cet axe a également concerné l'étude du test et du diagnostic dans les systèmes à base de réseaux de capteurs sans fil. Les caractéristiques communes avec les systèmes RFID, notamment l'inclusion d'un grand nombre d'équipements communicants au comportement incertain, ont permis d'approfondir la compréhension de ce type de systèmes. Les résultats obtenus dans le cadre des systèmes à base de réseaux de capteurs ont concerné le niveau des données, le niveau des nœuds capteurs ainsi que le niveau applicatif permettant ainsi une prise en compte globale de ce type de systèmes.

Les travaux sur les réseaux de capteurs se sont déroulés dans le cadre du projet CEDRE avec le laboratoire LASTRe de l'Université Libanaise et dans le cadre du projet ArtEco avec le laboratoire LIG de Grenoble. Ils ont fait l'objet de la thèse de Dima Hamdan.

V.6.2. Perspectives

Les travaux menés dans le cadre des systèmes RFID et des systèmes à base de réseaux de capteurs ont eu pour objectif commun de prendre en compte la diversité des composants présents dans de tels systèmes : composants matériels, composants logiciels, composants de communication. Ces composants sont souvent agencés sous forme de couches ou de niveaux et la plupart des travaux existants se sont intéressés principalement à un niveau donné, tout en ignorant les autres niveaux. Dans nos travaux, une première approche de prise en compte des interactions entre ces différents niveaux a été menée.

Bien que les résultats obtenus soient intéressants, ils sont relativement préliminaires et de nombreux aspects peuvent être approfondis.

D'une part, les perspectives de cet axe rejoignent celles du premier axe présentées dans le Chapitre 1 pour ce qui concerne l'approfondissement de l'étude des interactions possibles entre le matériel et le logiciel. Les équipements matériels présentent parfois une intelligence propre pouvant être exploitée pour l'autotest de ces équipements, en lien avec la couche de

l'intergiciel, dont le rôle est central. A ce titre, il serait également intéressant de prendre en compte les défaillances pouvant survenir dans cette couche en intégrant par exemple les travaux menés sur le test et le diagnostic des composants logiciels.

Les principaux outils utilisés dans nos travaux sont le test en ligne (pouvant prendre une forme de surveillance, ou de test statistique) et le diagnostic selon une approche système (probabiliste ou déterministe). De tels outils se sont avérés utiles pour les objectifs fixés, mais peuvent être limitatifs. A l'avenir, d'autres outils pourront être davantage explorés. En effet, l'utilisation croissante de l'Internet des Objets dans de nombreuses applications entraîne une complexité croissante des applications. Cette complexité se manifeste notamment à deux niveaux :

- La grande quantité de données transmises entre les équipements et devant être gérées,
- La structure des applications même qui tend à être hiérarchique en vue d'en simplifier la gestion.

Ce préoccupations rejoignent un domaine en plein essor qui est celui de l'informatique en nuage (ou *cloud computing*, incluant notamment le *cloud mobile* ou le cloud à base de réseaux de capteurs (*sensor cloud*) qui nous intéressent directement). Les travaux sur le test de ce type d'applications s'orientent vers une approche de type service [136], permettant la mise en commun des ressources de test. Ainsi la contrainte d'adaptation des services de test aux applications est de plus en plus forte et nécessite de déterminer le niveau précis. Ceci pourrait constituer une perspective intéressante de nos travaux[29].

[29] Cette perspective de recherche devrait être approfondie dans le cadre d'un projet de CRCT (Congé pour Recherches ou Conversions Thématiques) actuellement en cours de demande et se déroulerait dans le cadre d'une collaboration avec le Professeur Gao de l'Université de San José en Californie, durant l'année 2014-15.

CONCLUSION GENERALE

Ce mémoire a présenté les recherches menées depuis une dizaine d'années dans le domaine de la sûreté de fonctionnement des systèmes embarqués et répartis.

Ces travaux se caractérisent par la volonté de prendre en compte tant les composants matériels que logiciels utilisés dans ce type de systèmes.

Dans le premier chapitre, les différents travaux menés sur le test et le diagnostic en ligne des composants logiciels sont décrits ainsi que les perspectives de cet axe, allant principalement vers une prise en compte des interactions entre les composants logiciels et matériels pour des approches plus efficaces.

Le second chapitre a montré une étude de cas globale sur un type de systèmes dont l'utilisation est en plein essor : les systèmes RFID, qui sont une des formes que prend l'Internet des Objets, sujet de travaux et de recherches en cours. Cet axe nous a surtout permis d'analyser un type de système présentant un comportement aléatoire en raison des défauts pouvant survenir dans le système même, ou dans son environnement. Le travail d'analyse, de modélisation, de prototypage et d'évaluation qui a accompagné cette étude nous a permis d'une part d'élargir nos compétences et d'autres part de nous ouvrir des perspectives de recherche en lien avec les évolutions récentes des systèmes, notamment l'informatique en nuage.

L'ensemble de ces travaux a été possible grâce à l'encadrement de quatre thèses de Doctorat, résumées ci-dessous :

- Thèse de Rafik Kheddam sur le thème « Approches logicielles de sûreté de fonctionnement pour les systèmes RFID », soutenue le 9 avril 2014.
- Thèse de Dima Hamdan, en co-tutelle avec l'Université Libanaise, sur le thème « Détection et diagnostic des fautes dans des systèmes à base de réseaux de capteurs sans fil », soutenue le 20 février 2013.
- Thèse de Gilles Fritz sur le thème « Simulation de fautes pour l'évaluation du test en ligne de systèmes RFID », soutenue le 10 décembre 2012.
- Thèse de Thi-Quynh Bui sur le thème « Service de diagnostic en ligne pour les applications à base de composants logiciels », soutenue le 14 octobre 2009.

Différents projets ont également permis de faire avancer ces travaux. Au niveau international, il y a eu le projet Européen EURNEX sur la sûreté et la sécurité dans les

applications ferroviaires et le projet CEDRE sur la tolérance aux fautes dans les applications à base de réseaux de capteurs. Au niveau national, les principaux projets menés sont : le projet ANR SafeRFID sur la sûreté de fonctionnement dans les systèmes RFID, le projet ArtEco en collaboration avec le laboratoire LIG sur la qualité de service dans les applications à base de réseaux de capteurs et le projet de collaboration avec l'INRETS de Lille sur le diagnostic dans les réseaux embarqués utilisés dans les applications ferroviaires.

BIBLIOGRAPHIE

[1] R. Isemann, Fault-Diagnosis Systems, An Introduction from Fault Detection to Fault Tolerance, Springer, 2006.

[2] M. Barborak, M. Malek et A. Dahbura, «The consensus problem in fault-tolerant computing,» *ACM Computing Surveys,* vol. 25, n°2, pp. 171-220, juin 1993.

[3] O. Benkahla, C. Aktouf et C. Robach, «Diagnostic des architectures parallèles : un état de l'art,» *Technique et Science Informatiques,* vol. 16, n°2, pp. 195-224, février 1997.

[4] S. Lee et K. G. Shin, «Probabilistic Diagnosis of Multiprocessor Systems,» *ACM Computing Surveys,* vol. 26, n° 1, pp. 121-139, mars 1994.

[5] G. T. Heineman et W. T. Council, Component-based software engineering: putting the pieces together, Addison-Wesley Professional, 2001.

[6] I. Crnkovic et M. Larsson (Editors), Building Reliable Component-Based Software Systems, Artech House Publishers, 2002.

[7] N. R. May, H. W. Schmidt et I. E. Thomas, «Service Redundancy Strategies in Service-Oriented Architectures,» *35th Euromicro Conference on Software Engineering and Advanced Applications*, Patras, Grèce, 27-29 août 2009.

[8] A. Diaconescu, «Automatic Performance Optimisation of Component-Based Entreprise systems via Redundancy,» Thèse de Doctorat, Dublin, Avril 2006.

[9] DECOS, «http://www.decos.at/,» [En ligne].

[10] CLEOPATRE, «http://cleopatre.rts-software.org,» [En ligne].

[11] J. Fraga, F. Siqueira et F. Favarim, «An Adaptive Fault-Tolerant Component Model,» *9th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS'03 Fall)*, Capri, Italie, 1-3 octobre 2003.

[12] D. Hagimont et V. Marangozova, «An Infrastructure for CORBA Component Replication,» *1st IFIP/ACM Working Conference on Component Deployment*, Berlin, Allemagne, juin 2002.

[13] D. Hagimont et V. Marangozova, «Non-Functional Replication Management in the CORBA Component Model,» *8th International Conference on Object-Oriented, Information Systems*, 2002.

[14] K. S. da Gama, «Vers les applications fiables basées sur des composants dynamiques,» Thèse de Doctorat de l'Université de Grenoble, 6 octobre 2011.

[15] I. Crnković, S. Sentilles, A. Vulgarakis et M. R. V. Chaudron, «A Classification Framework for Software Component Models,» *IEEE Transactions on Software Engineering,* vol. 37, n°5, pp. 593 - 615, septembre-octobre 2011.

[16] C. Szyperski, Component Software: Beyond Object-Oriented Programming, Addison-Wesley, ACM Press, Component Software Series, 2002.

[17] CCM, «http://www.omg.org/spec/CCM/,» [En ligne].

[18] EJB, «http://docs.oracle.com/cd/E24329_01/web.1211/e24446/ejbs.htm,» [En ligne].

[19] Fractal, «http://fractal.ow2.org/documentation.html,» [En ligne].

[20] R. van Ommering, F. van der Linden, J. Kramer et J. Magee, «The Koala component model for consumer electronics software,» *Computer,* vol. 33, n°3, pp. 78-85, 2000.

[21] AUTOSAR, «http://www.autosar.org/,» [En ligne].

[22] OPC, «http://www.opcfoundation.org/,» [En ligne].

[23] Open Service Gateway initiative, «http://www.osgi.org/,» [En ligne].

[24] H. Maaskant, «A Robust Component Model for Consumer Electronic Products,» *ser. Philips Research. Springer,* vol. 3, n° 167_192, 2005.

[25] T. Q. Bui, «Un service de diagnostic en ligne de composants logiciels,» Thèse de Doctorat de Grenoble INP, 14 octobre 2009.

[26] F. P. Preparata, G. Metze et R. T. Chien, «On the Connection Assignment Problem of Diagnosable Systems,» *IEEE Transactions on Electronic Computers,* vol. EC-16, n° 6, pp. 848-854, 1967.

[27] G.-H. Hsu et J. Tan, «A Local Diagnosability Measure for Multiprocessor Systems,» *IEEE Transactions on Parallel and Distributed Systems,* vol. 18, n° 5, pp. 598-607, mai 2007.

[28] S.-Y. Hsieh et Y.-S. Chen, «Strongly Diagnosable Systems under the Comparison Diagnosis Model,» *IEEE Transactions on Computers,* vol. 57, n° 12, pp. 1720-1725, décembre 2008.

[29] M. Mánik et E. Gramatová, «Diagnosis of Faulty Units in Regular Graphs under the PMC Model,» *12th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, Liberec, République Tchèque, Avril 2009.

[30] T.-L. Kung, H.-C. Chen et J. Tan, «On the Faulty Sensor Identification Algorithm of Wireless Sensor Networks under the PMC Diagnosis Model,» *6th International Conference on Networked Computing and Advanced Information Mangement (NCM'10)*, Séoul, Corée du Sud, Août 2010.

[31] N.-W. Chang et S.-Y. Hsieh, «Conditional Diagnosability of Augmented Cubes under the PMC Model,» *IEEE Transactions on Dependable and Secure Computing,* vol. 9, n° 1, pp. 46-60, janvier-février 2012.

[32] P. L. Lai, «A Systematic Algorithm for Identifying Faults on Hypercube-like Networks under the Comparison Model,» *IEEE Transactions on Reliability,* vol. 61, n° 2, pp. 452-459, juin 2012.

[33] C.-K. Lin, T.-L. Kung et J. J. Tan, «An Algorithmic Approach to Conditional-Fault Local Diagnosis of Regular Multiprocessor Interconnected Systems under the PMC Model,» *IEEE Transactions on Computers,* vol. 62, n° 3, pp. 439-451, mars 2013.

[34] P.-L. Lai, M.-Y. Chiu et C.-H. Tsai, «Three Round Adaptive Diagnosis in Hierarchical Multiprocessor Systems,» *IEEE Transactions on Reliability,* vol. 62, n° 3, pp. 608-617, septembre 2013.

[35] B. Beizer, Software testing techniques, New York: Van Nostrand Reinhold, 2nd Edition, 1990.

[36] S. Beydeda et V. Gruhn, «State of the art in testing components,» *Third International Conference on Quality Software*, Dallas, Texas, USA, 6-7 novembre 2003.

[37] E. Martins, C. Toyota et R. Yanagawa, «Constructing self-testable software components,» *International Conference on Dependable Systems and Networks, DSN 2001*, Goteborg, Suède, 1-4 juillet 2001.

[38] Y. Wang, G. King et H. Wickburg, «A method for built-in tests in component-based software maintenance,» *Third European Conference on Software Maintenance and Reengineering*, Amsterdam, Pays-Bas, 3-5 mars 1999.

[39] Y. Wang, G. King, M. Fayad, D. Patel, I. Court, G. Staples et M. Ross, «On built-in test reuse in object-oriented framework design,» *ACM Computing Surveys,* vol. 32, n° 1es, mars 2000.

[40] C. Atkinson et H.-G. Groß, «Built-in contract testing in model-driven, component-based development,» *ICSR Workshop on Component-Based Development Processes*, 2002.

[41] J. Hörnstein et H. Edler, «Test reuse in CBSE using built-in tests,» *Workshop on component-based Software Engineering, Composing systems from components*, 2002.

[42] M. Momotko et L. Zalewska, «Component+ Built-In Testing - A Technology for Testing Software Components,» *Foundations of Computing and Decision Sciences,* vol. 29, n° 1-2, pp. 133-148, 2004.

[43] J. Z. Gao, K. K. Gupta, S. Gupta et S. S. Y. Shim, «On Building Testable Software Components,» *First International Conference on COTS-Based Software Systems (ICCBSS '02)*, 2002.

[44] M. Jaffar-ur Rehman, F. Jabeen et A. Polini, «Testing software components for integration: a survey of issues and techniques,» *Software Testing, Verification & Reliability,* vol. 17, n° 2, pp. 95-133, Juin 2007.

[45] Y. Wu, M.-H. Chen et J. Offutt, «UML-based Integration Testing for Component-Based Software,» *2nd International Conference on COTS-Based Software Systems (ICCBSS 2003)*, Ottawa, Canada, février 2003.

[46] L. Gallagher, J. Offutt et A. Cincotta, «Integration Testing of Object-Oriented Components Using Finite State Machines,» *Software Testing, Varification and Reliability,* vol. 16, n° 4, pp. 211-263, 2006.

[47] J.-M. Voas, «Certifying Off-The-Shelf Software Components,» *IEEE Computer,* vol. 31, n° 6, pp. 53-59, 1998.

[48] A. Bertolino et A. Polini, «A Framework for Component Deployment Testing,» *25th International Conference on Software Engineering*, mai 2003.

[49] S. H. Edwards, «A Framework for Practical, Automated Black-Box Testing of Component-Based Software,» *Software Testing, Verification & Reliability,* vol. 11, n° 2, pp. 97-111, juin 2001.

[50] S. Beydeda et V. Gruhn, «The Self-Testing COTS Components (STECC) Strategy – A new form of improving component testability,» *ACTA Press*, USA, 2003.

[51] S. Beydeda, The Seft-Testing COTS Components (STECC) Method, Munich, Allemagne: Martin Meidenbauer Verlag, 2004.

[52] H.-G. Groß, C. Atkinson, F. Barbier, N. Belloir et J.-M. Bruel, Built-in contract testing for component-based development, Business Component-Based Software Engineering, Springer US, 2003, pp. 65-82.

[53] E. Nishijima, H. Yamamoto, K. Kawano, K. Fujiwara et K. Oshima, «On-line testing for application software of widely distributed system,» *15th Symposium on Reliable Distributed Systems*, Niagara-on-the-Lake, Ontario, Canada, 23-25 octobre 1996.

[54] «http://www.ist-plastic.org/,» [En ligne].

[55] D. Niebuhr, A. Rausch, C. Klein, J. Reichmann et R. Schmid, «Achieving Dependable Component Bindings in Dynamic Adaptive Systems - A Runtime Testing Approach,» *Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems, SASO '09*, San Francisco, CA, USA, 14-18 septembre 2009.

[56] P. H. Deussen, G. Din et I. Schieferdecker, «An On-Line Test Platform for Component-Based Systems,» *27th Annual NASA Goddard Software Engineering Workshop (SEW-27'02)*, 2002.

[57] Y. Yao et Y. Wang, «A Framework for Testing Distributed Software Components,» *Canadian Conference on Electrical and Computer Engineering*, Saskatoon, Canada, mai 2005.

[58] «http://www.componentworld.nu/,» [En ligne].

[59] N. Q. Tran, «Study of Automatic Test for Application to Components in the On-Line Context,» Rapport de stage de M2R de l'Université de Grenoble, septembre 2011.

[60] D. Lee et M. Yannakakis, «Principles and Methods of Testing Finite State Machines - A Survey,» *Proceedings of the IEEE,* vol. 84, n° 8, p. 1090–1123, août 1996.

[61] A. T. Dahbura, K. K. Sabnani et W. J. Hery, «Spare Capacity as a Means of Fault Detection and Diagnosis in Multiprocessor Systems,» *IEEE Transactions on Computers,* vol. 38, n° 6, pp. 881-891, 1989.

[62] D. Suliman, B. Paech , L. Borner, C. Atkinson, D. Brenner, M. Merdes et R. Malaka, «The MORABIT Approach to Runtime Component Testing,» *30th Annual International Computer Software and Applications Conference (COMPSAC 2006)*, Chicago, USA, 17-21 septembre 2006.

[63] O. Aktouf et T. Q. Bui, «Diagnosis Service for Embedded Software Component based Systems,»

Second International Workshop on Engineering Fault Tolerant Systems (EFTS), Dubrovnik, Croatie, septembre 2007.

[64] T. Q. Bui, O. Aktouf et M. Dang, «Software Component Diagnosis Service: Architecture Description,» *Third International Symposium on Industrial Embedded Systems*, La Grande Motte, France, juin 2008.

[65] T. Q. Bui et O. Aktouf, «Inter-component testing for system-level diagnosis of embedded component based applications,» *7th Multidisciplinary International Conference on Quality and Reliability*, Tanger, Maroc, mars 2007.

[66] T. Q. Bui, O. Aktouf, L. D'orazio et M. Dang, «Service de Diagnostic pour Composants Logiciels,» *8ème Congrès international pluridisciplinaire en Qualité et Sûreté de Fonctionnement*, Besançon, France, mars 2009.

[67] T. Q. Bui, O. Aktouf, M. Dang, L. Gürgen et C. Roncancio, «Diagnosis Service for Software Component and its Application to a Heterogeneous Sensor Data Management System,» *Second International Conference on Dependability*, Athènes, Grèce.

[68] D. Gizopoulos, M. Psarakis, M. Hatzimihail, M. Maniatakos, A. Paschalis, A. Raghunathan et S. Ravi, «Systematic Software-Based Self-Test for Pipelined Processors,» *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 16, n° 11, pp. 1441-1453, 2008.

[69] D. Gizopoulos, «Online Periodic Self-Test Scheduling for Real-Time Processor-Based Systems Dependability Enhancement,» *IEEE Transactions on Dependable and Secure Computing,* vol. 6, n° 2, pp. 152-158, 2009.

[70] A. Pellegrini et V. Bertacco, «Application-aware diagnosis of runtime hardware faults,» *International Conference on Computer-Aided Design*, 17-19 novembre 2010.

[71] L. Lednicki, A. Petricic et M. Zagar, «A Component-Based Technology for Hardware and Software Components,» *35th Euromicro Conference on Software Engineering and Advanced Applications (SEAA),*, Pataras, Grèce, 2009.

[72] K. Ashton, «That 'Internet of Things' thing, RFID Journal,» 22 juin 2009. [En ligne]. Available: http://www.rfidjournal.com/articles/view?4986.

[73] «Projet ANR SafeRFID,» [En ligne]. Available: http://www.agence-nationale-recherche.fr/projet-anr/?tx_lwmsuivibilan_pi2[CODE]=ANR-10-JCJC-0305.

[74] G. Fritz, «Simulation de fautes pour l'évaluation du test en ligne de systèmes RFID,» Thèse de Doctorat de l'Université de Grenoble, 10 décembre 2012.

[75] R. Kheddam, «Approches logicielles de sûreté de fonctionnement pour les systèmes RFID,» Thèse de Doctorat de l'Université de Grenoble, 9 avril 2014.

[76] A. Villemeur, Sûreté de fonctionnement des systèmes industriels, Edition Eyrolles, 1997.

[77] G. Fritz, B. Maaloul, V. Beroulle, O.-E.-K. Aktouf et D. Hély, «Read rate profile monitoring for defect detection in RFID Systems,» *IEEE RFID-Technologies and Applications (RFID-TA)*, Sitges, Espagne, 15-16 septembre 2011.

[78] G. Fritz, V. Beroulle, O. Aktouf et D. Hély, «SystemC Modeling of RFID Systems for Robustness Analysis,» *IEEE 19th International Conference on Software Telecommunications and Computer Networks (SoftCOM), Symposium on RFID Technology and Applications (RFID-TA)*, 15-16 septembre 2011.

[79] R. Kheddam, O. Aktouf et I. Parissis, «SafeRFID-MW: a RFID Middleware with runtime fault diagnosis,» *Journal of Communications Software and Systems,* vol. 9, n° 1, pp. 57-73, mars 2013.

[80] R. Kheddam, O. Aktouf, I. Parissis et S. Boughazi, «Monitoring of RFID Failures Resulting from LLRP Misconfigurations,» *21st IEEE International Conference on Software, Telecommunications and Computer Networks*, Split, Croatie, 2013.

[81] S. Boughazi, «Conception d'une solution d'analyse des flux de données RFID en vue du diagnostic de fautes,» Rapport de stage de M2R de l'Université Paul Sabatier Toulouse 3, septembre 2013.

[82] D. Paret, RFID en ultra et super hautes fréquences – UHF-SHF – Théorie et mise en œuvre, Dunod, 2008.

[83] Gartner's Hype Cycle Special Report for 2005, Août 2005.

[84] M. Desertot, C. Marin et D. Donsez, «SensorBean : Un modèle à composants pour les services basés capteurs,» *4ème Conférence Francophone autour des Composants Logiciels*, Le Croisic, France, 6-8 Avril 2005.

[85] «Etiquettes électroniques (RFID). Infrastructures logicielles et middleware,» DGE-RFID-GT-Middleware, avril 2006 (révision 1.0).

[86] R. Shorey et et al., Mobile, Wireless and Sensor Networks : Technology, Applications and Future Directions, Chapitre «WinRFID – A middleware for the enablement of Radio Frequency Identification (RFID) based Applications» B. S. Prabhu et al., John Wiley and Sons Inc., 2006.

[87] W. Rudametkin et et al., «NFCMuseum: an Open-Source Middleware for Augmenting Museum Exhibits,» *International Conference on Pervasive Services, IPCS'08*, Italie, juin 2008.

[88] N. Kefalakis, J. Soldatos, K. Gama et D. Donsez, «Supply Chain Management and NFC Picking Demonstrations using the AspireRFID Middleware Platform,» *Middleware*, 2008.

[89] M. E. Ajana, H. Harroud, M. Boulmalf et H. Hamam, «FlexRFID: A Flexible Middleware for RFID Applications Development,» IEEE Press, IFIP International Conference on Wireless and Optical Communications Networks, WOCN '09., Caire, Egypte, 2009.

[90] A. Sengupta et S. Z. Schiller, «FlexRFID: A design, development and deployment framework for RFID-based business applications,» *Information Systems Frontiers,* vol. 12, n° 5, pp. 551-562, November 2010.

[91] L. Catarinucci, R. Colella et L. Patrono, «On the use of passive UHF RFID tags in the pharmaceutical supply chain: a novel enhanced tag versus hi-performance commercial tags,» *International Journal of Radio Frequency Identification Technology and Applications (IJRFITA),* vol. 4, n° 2, 2013.

[92] P. Krishna et D. Husak, «RFID Infrastructure,» *IEEE Communications Magazine,* vol. 45, n° 9, pp. 4-10, septembre 2007.

[93] ISO/IEC 7498-1 – Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model.

[94] V. Derbek, C. Steger, R. Weiss, D. Wischounig, J. Preishuber-Pfluegl et M. Pistauer, «Simulation Platform for UHF RFID,» *Design, Automation & Test in Europe Conference & Exhibition*, 16-20 avril 2007.

[95] J. Wang et D. Yang, «Design of a Multi-Protocol RFID Tag Simulation Platform Based on Supply Chain,» *International Conference on Management and Service Science*, 20-22 septembre 2009.

[96] «RIFIDI,» [En ligne]. Available: http://www.rifidi.org.

[97] L. Mirowski, J. Hartnett et R. Williams, «Tyrell: A RFID simulation platform,» *5th International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 7-10 décembre 2009.

[98] H. Woellik, «A Simulation tool for RFID anti-collision algorithms based on ALOHA,» *16th International Conference on Systems, Signals and Image Processing*, 18-20 juin 2009.

[99] C. Angerer et R. Langwieser, «Flexible evaluation of RFID system parameters using rapid prototyping,» *IEEE International Conference on RFID*, 27-28 avril 2009.

[100] C. Angerer, B. Knerr, M. Holzer, A. Adalan et M. Rupp, «Flexible simulation and prototyping for RFID designs,» *First international EURASIP Workshop on RFID Technology*, sptembre 2007.

[101] C. Floerkemeier et R. Pappu, «Evaluation of RFIDSim – a physical and Logical,» *IEEE International Conference on RFID*, 16-17 avril 2008.

[102] C. Floerkemeier et S. Sarma, «RFIDSim – a physical and Logical layer simulation engine,» *IEEE Transactions on Automation Science and Engineering,* vol. 6, n° 1, pp. 33-43, janvier 2009.

[103] B. Glover et H. Bhatt, RFID Essentials, O'Reilly, 2006.

[104] P. Sanghera, F. Thornton, B. Haines, F. Kung Man Fung, J. Kleinschmidt, A. M. Das, H. Bhargava et A. Campbell, How to Cheat at Deploying and Securing RFID, Syngress Publishing, Inc. Elsevier, Inc., 2007.

[105] R. Derakhshan, M. E. Orlowska et X. Li, «RFID Data Management: Challenges and Opportunities,» IEEE International Conference on RFID, USA, 2007.

[106] L. Yang, J. Cao, W. Zhu and S. Tang, "A Hybrid Method for achieving High Accuracy and Efficiency in Object Tracking using Passive RFID," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Lugano, pp. 109-115, March 2012.

[107] R. Rakotomalala, Tests de normalité – Techniques empiriques et tests statistiques, Document de cours, Université Lumière Lyon 2, 1er octobre 2011.

[108] G. Fritz, V. Beroulle, O.-E.-K. Aktouf, M.-D. Nguyen et D. Hély, «RFID system on-line testing based on the evaluation of the tags Read-Error-Rate,» *IEEE, Mixed-Signals, Sensors and Systems Test Workshop,* pp. 1-10, 2010.

[109] G. Fritz, V. Beroulle, O. Aktouf et D. Hély, «Evaluation of a new RFID system performance monitoring approach,» *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 12-16 mars 2012.

[110] G. Fritz, V. Beroulle, O.-E.-K. Aktouf, M. D. Nguyen et D. Hély, «RFID System On-line Testing Based on the Evaluation of the Tags Read-Error-Rate,» *Journal of Electronic Testing (JETTA),* vol. 27, pp. 267-276, Juin 2011.

[111] S. Rangarajan et D. Fussell, «A Probabilistic Method for Fault Diagnosis of Multiprocessor Systems,» *IEEE 18th International Symposium on Fault-Tolerant Computing*, Tokyo, Japan, 1988.

[112] D. Fussell et S. Rangarajan, «Probabilistic diagnosis of multiprocessor systems with arbitrary connectivity,» *IEEE 19th International Symposium on Fault-Tolerant Computing, FTCS-19. Digest of Papers.*, Chicago, IL, pp. 560-565, 1989.

[113] M. Kobayashi, T. Takabatake, T. Matsushima et S. Hirasawa, «Probabilistic fault diagnosis and its analysis in multicomputer systems,» *2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Alaska, USA, 9-12 octobre 2011.

[114] «EPCglobal,» [En ligne]. Available: www.epcglobalinc.org.

[115] C. Floerkemeier, C. Roduner et M. Lampe, «RFID Application Development With the Accada Middleware Platform,» *IEEE Systems Journal,* pp. 1-13, décembre 2007.

[116] C. Floerkemeier, M. Lampe et C. Roduner, «Facilitating RFID Development with the Accada Prototyping Platform,» *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops. PerCom Workshops '07.*, Zurich, Suisse, 2007.

[117] J. Soldatos, "AspireRFID Can Lower Deployment Costs," 16 Mars 2009. [Online]. Available: http://www.rfidjournal.com/article/view/4661.

[118] «ASPIRE Project, "ASPIRE - The EU funded project that brings RFID to SMEs",» [En ligne]. Available: http://www.fp7-aspire.eu.

[119] Sun Microsystems, Inc., "Sun Java™ System RFID Software 3.0 Administration Guide," February 2006.

[120] Sun Microsystems, Inc., "Sun Java™ System RFID Software 3.0 Developer's Guide," February 2006.

[121] «WinRFID,» [En ligne]. Available: www.wireless.ucla.edu/rfid/winrfid.

[122] B. Prabhu, X. Su, C. Qiu, H. Ramamurthy, P. Chu et R. Gadh, «WinRFID – Middleware for Distributed RFID Infrastructure,» International Workshop on Radio Frequency Identification (RFID) and Wireless Sensors, Kanpur, Inde, 11-13 novembre 2005.

[123] B. S. Prabhu, X. Su, H. Ramamurthy, H.-C. Chu et R. Gadh, «WinRFID – A Middleware for the enablement of Radio Frequency Identification (RFID) based Applications,» *in Mobile, Wireless and Sensor Networks: Technology, Applications and Future,* pp. 331-336, 2005.

[124] «UCLA WINMEC, "RFID@WINMEC - RFID Research," 2005,» [En ligne]. Available:

http://www.winmec.ucla.edu/rfid/winrfid.

[125] J. Chamberlain, C. Blanchard, S. Burlingame, S. Chandramohan, E. Forestier, G. Griffith, M. L. Mazzara, S. Musti, S.-I. Son, G. Stump et C. Weiss, IBM WebSphere RFID Handbook : A Solution Guide, IBM, 2006.

[126] A. P. Anagnostopoulos, J. K. Soldatos et S. G. Michalakos, «REFiLL: A lightweight programmable middleware platform for cost effective RFID application development,» *Pervasive and Mobile Computing,* vol. 5, n° 1, pp. 49-63, février 2009.

[127] N. Ahmed, «Reliable Framework for Unreliable RFID Devices,» 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Mannheim, 2010.

[128] N. Ahmed, R. Kuma, R. S. French et U. Ramachandran, «RF²ID : A Reliable Middleware Framework for RFID Deployment,» *International IEEE Parallel and Distributed Processing Symposium*, Long Beach, CA, mars 2007.

[129] N. Ahmed et R. Umakishore, «A Path Based Reliable Middleware Framework for RFID Devices,» IEEE, Atlanta, USA, 2010.

[130] R. Kheddam, O. Aktouf et I. Parissis, «An extended LLRP model for RFID system test and diagnosis,» *8th Workshop on Advances in Model Based Testing, dans le cadre de 5th IEEE International Conference on Software Testing, Verification and Validation*, Montreal, Canada, 17-21 avril 2012.

[131] R. Kheddam, O. Aktouf et I. Parissis, «Online Monitoring and Diagnosis of RFID Readers and Tags,» *20th IEEE International conference on Software, Telecommunications and Computer Networks*, Split, Croatie, 11-13 septembre 2012.

[132] D. Hamdan, «Détection et diagnostic des fautes dans des systèmes à base de réseaux de capteurs sans fil,» Thèse de Doctorat de l'Université de Grenoble et de l'Université Libanaise, 20 février 2013.

[133] D. Hamdan, O. Aktouf, I. Parissis, B. El Hassan et A. Hijazi, «Online Data Fault Detection for Wireless Sensor Networks - Case Study,» *IEEE. Third International Conference on Wireless Communication in Unusual and Confined Areas*, Clermont-Ferrand, France, août 2012.

[134] D. Hamdan, O. Aktouf, I. Parissis, B. EL Hassan et A. Hijazi, «A Self monitoring, Adaptative and Resource Efficient Approach for Improving QoS in Wireless Sensor Networks,» *IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Sanya, Chine, 10-12 octobre 2012.

[135] D. Hamdan, O. Aktouf, I. Parissis, A. Hijazi, M. Sarkis et B. EL Hassan, «SMART Service for Fault-Diagnosis in Wireless Sensor Networks,» *IEEE 6th International Conference on Next Generation Mobile Applications, Services and Technologies*, Paris, France, 12-14 septembre 2012.

[136] J. Gao, X. Bai et W.-T. Tsai, «Testing as a Service (TaaS) on Clouds,» *2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE)*, Redwood City, USA, 25-28 mars 2013.

TABLE DES MATIÈRES

Sommaire 3
Remerciements 5
Introduction 7
Chapitre 1 Test et Diagnostic en ligne de composants logiciels embarqués 9
I. Positionnement de l'étude 9
II. Composants logiciels 11
II.1. Définitions et caractéristiques 11
II.2. Modèle considéré de composants logiciels 13
III. L'approche de diagnostic système à partir de test inter-composants 13
III.1. Définitions 13
III.2. Principe du diagnostic centralisé 14
III.3. Principe du diagnostic distribué 15
IV. Test de composants logiciels 16
IV.1. Etapes et difficulté du test de composants 16
IV.2. Test d'intégration ou de déploiement 18
IV.3. Test en ligne de composants logiciels 23
IV.4. Conclusion 24
V. Contributions 25
V.1. Niveau d'implémentation 25
V.2. Le service DISCO (DIagnosis Service for Component-based applications) 26
VI. Bilan et perspectives 30
VI.1. Bilan 30
VI.2. Perspectives : Unification matérielle/logicielle 30
Chapitre 2 Test et diagnostic en ligne des systèmes RFID 33
I. Positionnement de l'étude 33
I.1. Introduction 33
I.2. Méthodologie de travail 34
II. Description des systèmes RFID 36
II.1. Principaux composants des systèmes RFID 36
II. 2. Les fonctions d'un middleware RFID 38
III. Analyse, modélisation et simulation des systèmes RFID 39

III.1. AMDE du système tag-lecteur 40
III.2. Modèles de fautes pour le couple tag-lecteur 41
III.3. Simulation des systèmes RFID 42
IV. Test en ligne et injection de fautes 47
IV.1. Les approches existantes 47
IV.2. L'approche de test statistique par étude du profil de lecture 48
IV.3. Evaluation de la méthode Profil 50
V. Intégration du test et du diagnostic au niveau de l'intergiciel RFID 51
V.1. Existant en sûreté de fonctionnement et tolérance aux fautes dans les intergiciels RFID 52
V.2. Etude AMDE et modèles de fautes 57
V.3. Diagnostic probabiliste dans les systèmes RFID 58
V.4. Evaluation 62
V.5. Autres travaux 64
V.6. Bilan et perspectives 64
Conclusion générale 67
Bibliographie 69
Table des Matières 79
Annexes 81

ANNEXES

Annexe A : Curriculum Vitae

Annexe B : Articles joints

Annexe A :Curriculum Vitæ

Mme Oum-El-Kheir Aktouf

Grenoble INP - Esisar
Laboratoire LCIS

Section du CNU : 27 (Informatique)

Sommaire

1. Informations générales 83

2. Activités de recherche 85

3. Activités d'enseignement 92

4. Charges collectives et d'intérêt général 96

5. Publications 97

1. Informations générales

1.1. État civil

Nom :	Benkahla épouse Aktouf
Prénom :	Oum-El-Kheir
Situation familiale :	Mariée, 2 enfants
Nationalité :	Française
Adresse personnelle :	Lot Les Buissonnets 2K Allée du Commandant Teinturier 26100 Romans-sur-Isère
Fonction actuelle :	Maître de Conférences en Informatique à Grenoble INP – Esisar, affectée en recherche au laboratoire LCIS
Adresse professionnelle :	Esisar - Grenoble INP 50, rue Barthélémy de Laffemas, BP 54 26902 Valence Cedex 9, France
Téléphone :	04 75 75 94 46
Télécopie :	04 75 75 94 50
Adresse électronique :	Oum-El-Kheir.Aktouf@esisar.grenoble-inp.fr

1.2. Synopsis

Depuis Septembre 1999 :	Maître de Conférences à Grenoble INP – Esisar, titularisée au 1er septembre 2000
Septembre 1998 – Août 1999 :	Chercheur contractuel au laboratoire LCIS - Grenoble INP
Avril 1998 :	Qualification aux fonctions de Maître de Conférences au titre de la section 27
Septembre 1997 - Août 1998 :	ATER à l'Université Joseph Fourier, Grenoble I
Septembre 1996 - Août 1997 :	ATER à l'Université Pierre Mendès France, Grenoble II
Décembre 1993 - Juin 1997 :	Préparation du Doctorat d'Informatique Institut National Polytechnique de Grenoble Bourse de coopération franco-algérienne
Septembre 1992 - Juin 1993 :	Préparation d'un DEA (Diplôme d'Études Approfondies) d'Informatique Institut National Polytechnique de Grenoble (ENSIMAG) Option «Architecture, Parallélisme et Systèmes Répartis»
Septembre 1987 - Juin 1992 :	Études d'Ingénieur en Informatique Université d'Annaba (Algérie) Option «Systèmes Informatiques» Mention Bien
Juin 1987 :	Baccalauréat série Mathématiques Mention Très Bien

1.3. Titres universitaires

Doctorat d'Informatique, Grenoble INP

Titre du mémoire :	Diagnostic des Architectures Parallèles à Passage de Messages
Date de soutenance :	6 Juin 1997
Directeur :	Mme Chantal Robach, actuellement Professeur à Grenoble INP
Président du jury :	M. Guy Mazaré, Professeur à Grenoble INP
Rapporteurs :	M. Christian Landrault, DR LIRMM-CNRS, Montpellier
	M. Traian Muntean, Professeur à l'Université d'Aix-Marseille II
Examinateur :	M. Jean-Marc Vincent, Maître de Conférences à l'Université Grenoble I

DEA d'Informatique, Grenoble INP

Option :	Architecture, Parallélisme et Systèmes Répartis
Titre du mémoire :	Diffusion sous Contraintes Temporelles dans les Architectures Massivement Parallèles
Responsable :	M. Traian Muntean, actuellement Professeur à l'Université de la Méditerranée, Aix-Marseille II

Diplôme d'Ingénieur d'État en Informatique, Université d'Annaba, Algérie

Option :	Systèmes Informatiques
Titre du mémoire :	Interprétation et Mise au Point des Systèmes Systoliques
Responsable :	M. Sélim Ghanemi, Maître de Conférences à l'Université d'Annaba, Algérie

2. Activités de recherche

Mes activités de recherche ont pour principaux mots-clés la *sûreté de fonctionnement* et la *validation* des systèmes et ont essentiellement été menées dans une optique de réduction des coûts liés à la sûreté de fonctionnement et de développement de méthodes efficaces. Ainsi, j'ai tout d'abord travaillé durant ma thèse de doctorat sur la validation des architectures parallèles à passage de messages en utilisant des approches de diagnostic hors ligne. Par la suite, j'ai étendu mes compétences de diagnostic au logiciel et aux systèmes mixtes logiciels-matériels et ce, dans le cadre d'un contrat d'ATER avec l'Université Joseph Fourier, puis dans le cadre d'un contrat de recherches post-doctorales avec le laboratoire LCIS de Grenoble INP. Enfin, depuis ma prise de fonction en tant que Maître de Conférences à l'école d'Ingénieurs Esisar de Grenoble INP, je développe des approches de diagnostic en ligne pour les systèmes embarqués avec l'optique de considérer les différents composants de tels systèmes : logiciels, matériels, réseaux.

2.1. Sûreté de fonctionnement des systèmes embarqués (depuis l'année 2000)

Mots-clés : sûreté de fonctionnement, diagnostic en ligne, tests inter-composants, composants logiciels, systèmes logiciels-matériels, systèmes RFID, réseaux de capteurs.

Contexte

Depuis ma nomination en tant que Maître de Conférences, mes recherches se sont déroulées au laboratoire LCIS. Ce laboratoire, fondé en 1996, se caractérise par une forte pluridisciplinarité puisque les recherches qui y sont menées concernent tant la physique, l'automatique, l'électronique que l'informatique. Le thème fédérateur défini au début des années 2000 est « systèmes embarqués communicants ». Dans le cadre de ce thème global, mes activités de recherche portent plus spécifiquement sur la sûreté de fonctionnement et la tolérance aux fautes dans les systèmes embarqués. En effet, en raison des domaines d'utilisation de ce type de systèmes, souvent critiques, la sûreté de fonctionnement apparaît comme une problématique de toute première importance. Or, les solutions classiques de redondance structurelle peuvent ne pas être adaptées car elles nécessitent souvent un surcoût de ressources et d'énergie, éléments relativement limités au sein d'un système embarqué. Ceci a motivé l'orientation de mes travaux vers des solutions fondées principalement sur l'utilisation de *tests* et de *diagnostic en ligne*. Il m'est ensuite très vite apparu que prendre en compte la sûreté de fonctionnement globale d'un système embarqué nécessitait l'étude de ses différents composants (logiciels, matériels, réseaux) ainsi que des interactions qu'ils pouvaient avoir entre eux.

J'ai mené ces travaux tout d'abord au sein de l'équipe CSYS (Conception et sûreté de fonctionnement des SYStèmes distribués) dont j'ai assuré la responsabilité de septembre 2003 à septembre 2007. Cette équipe comportait 3 permanents dont les activités de recherche étaient complémentaires : *test des systèmes matériels*, *réseaux embarqués* et *diagnostic de système*. Cette complémentarité a permis de faire mûrir les activités en lien avec la sûreté de fonctionnement des systèmes embarqués. Deux thèses ont été soutenues[30] et trois projets de recherche ont été menés (le projet européen EURNEX et 2 projets dans le cadre du financement FITT[31]). Par la suite, une restructuration du laboratoire LCIS en septembre 2007 a permis de fusionner les travaux de l'équipe CSYS avec ceux de l'équipe ValSys (Validation des

[30] Il s'agit des thèses de O. Laouamri sur le test à distance des IP et de T.-Q. Bui sur le diagnostic des composants logiciels embarqués.
[31] Seuls les projets me concernant seront décrits par la suite.

Systèmes) dont la préoccupation principale était la validation des systèmes intégrés. Ce rapprochement a permis de mettre en commun des compétences complémentaires notamment sur les aspects matériels et logiciels, ce qui a donné lieu à 3 projets de recherche (SAFERFID, ARTeco et CEDRE décrits ci-dessous) et trois thèses (thèses de G. Fritz, R. Kheddam et D. Hamdan décrites ci-dessous).

Description

Depuis quelques années, l'évolution informatique a donné naissance à des systèmes de plus en plus sophistiqués, intégrant différents types de ressources (capteur, téléphone mobile, lecteur RFID, logiciel embarqué, *etc*.) Ces systèmes se caractérisent à des degrés divers par leur répartition, leur environnement variable et souvent hostile (perturbations électromagnétiques, milieu non conducteur bloquant la communication, *etc*.) et les contraintes de ressources et d'énergie. La problématique de la sûreté de fonctionnement de tels systèmes devient cruciale, d'autant plus qu'en raison de la nature des applications (surveillance et suivi à distance de malades, détection de substances toxiques dans des environnements industriels ou autres, inventaire de grands stocks d'articles, *etc*.) il est de plus en plus difficile et coûteux, voire impossible, d'avoir recours à l'intervention humaine en cas de dysfonctionnement.

Par ailleurs, la nature embarquée des différents éléments nécessite, pour une appréhension globale du système étudié, la prise en compte des composants matériels, logiciels et de communication (réseaux) qui forment le système étudié. En effet, les interactions entre ces différents types de composants peuvent influer sur la qualité globale du système.

Dans l'optique de réduire les coûts en ressources liés à la sûreté de fonctionnement, mais également dans le but de proposer des solutions de tolérance aux fautes « autonomes » répondant aux contraintes des nouvelles applications informatiques, je m'intéresse en particulier à des approches basées sur le *diagnostic de système*. Ce type de diagnostic utilise les ressources propres d'une application ou d'un système pour assurer la détection et la localisation des éléments défaillants, au moyen *de tests en ligne* et de *tests inter-composants*, conférant ainsi une certaine autonomie aux systèmes et applications. Toutefois, de nombreux verrous scientifiques et techniques doivent être levés, à l'image de la faisabilité des tests inter-composants en ligne, la définition de la nature des fautes traitées, l'analyse des résultats des tests pour en déduire l'état du système, la reconfiguration, le passage à l'échelle, *etc*.

Les domaines d'application de ces recherches concernent les applications à base de composants logiciels (projet DISCO), les systèmes RFID (projet SAFERFID) et les applications basées sur les réseaux de capteurs (projets ARTeco et CEDRE). Du point de vue régional, ils relèvent du projet Semba (Systèmes Embarqués) du cluster ISLE (Informatique, Signal, Logiciel Embarqué) ainsi que de l'ARC6 (Communauté de Recherches Académiques Rhône-Alpes). A l'international, ils m'ont permis de participer au projet européen (Réseau d'Excellence) EURNEX sur la sécurité dans le domaine ferroviaire.

Les principales contributions actuellement réalisées concernent :

- le développement d'un service de diagnostic en ligne pour les composants logiciels embarqués et son intégration dans une plate-forme de gestion de réseaux de capteurs,
- la modélisation et la proposition d'une première approche de test statistique adaptée aux systèmes RFID,
- le développement d'un simulateur pour les systèmes RFID (HF et UHF),
- les spécifications et le développement d'un prototype d'intergiciel RFID tolérant aux fautes permettant l'analyse des données issues d'étiquettes RFID en vue d'améliorer la sûreté de fonctionnement,

- le développement d'une approche de tolérance aux fautes multi-niveaux (données, nœuds capteurs, niveau applicatif) pour les systèmes à base de réseaux de capteurs sans fil,
- L'étude de faisabilité de tests inter-composants dans le cadre de réseaux embarqués utilisés dans les applications ferroviaires.

2.2. Validation de systèmes mixtes matériels-logiciels (de 1997 à 2000)

Mes recherches antérieures ont porté sur la validation des architectures massivement parallèles dans le cadre de la thèse de Doctorat (de 1994 à 1997) puis sur la validation de systèmes matériels-logiciels dans le cadre de recherches post-doctorales (de 1997 à 1999). Ces dernières recherches m'ont permis d'aborder plus particulièrement les aspects suivants :

- l'analyse de testabilité et le diagnostic du logiciel,
- le test et le diagnostic de haut niveau de systèmes matériels,
- l'unification des approches proposées pour une application aux systèmes mixtes logiciels-matériels.

Ces recherches ont été motivées par l'essor grandissant que connaissait la conception conjointe au milieu des années 90 et qui a soulevé le problème de la validation des systèmes mixtes logiciels-matériels. La démarche entreprise dans le cadre de ces recherches a été guidée par les observations suivantes :

- un système mixte comporte des composants logiciels et d'autres matériels qui interagissent à travers des interfaces ;
- grâce aux progrès de la CAO (Conception Assistée par Ordinateur), les composants matériels sont souvent décrits dans des langages de haut niveau, rendant ces descriptions assimilables à des programmes logiciels. Moyennant quelques efforts d'adaptation, il est alors envisageable d'appliquer les méthodes de validation du logiciel au matériel. Ceci ouvre la voie à des méthodes de validation unifiées, pouvant profiter directement aux systèmes logiciels-matériels.

2.3. Projets et collaborations de recherche en cours

Le projet SAFERFID

Ce projet concerne la sûreté de fonctionnement des systèmes RFID. Il est soutenu dans le cadre de l'ANR « Jeunes Chercheuses et Jeunes chercheurs ». Ce projet a débuté le 1er octobre 2010 et est prévu sur une durée de 42 mois. Le montant total de la subvention s'élève à 203,112 K€. Outre la coordination du projet, ce projet comporte quatre workpackages (WP) :

- WP1 : modélisation des systèmes RFID,
- WP2 : modélisation et simulation de fautes,
- WP3 : test et diagnostic en ligne,
- WP4 : amélioration de la tolérance aux fautes dans les systèmes RFID.

J'assure la responsabilité des WP2 et WP3. L'ensemble des WP comporte 15 tâches dont 8 sont sous ma responsabilité. L'équipe projet comprend 3 permanents du LCIS et 2 doctorants (Mr G. Fritz, Mr R. Kheddam et Mr O. Abdelmalek). Dans le cadre de ce projet, j'ai contribué au co-encadrement des thèses de G. Fritz et de R. Kheddam.

La thèse de G. Fritz, soutenue en décembre 2012, a concerné l'étude de la partie matérielle d'un système RFID (couple lecteur-étiquette) avec le développement d'un simulateur dont le but est de permettre l'étude de la tolérance aux fautes (analyse des modes de défaillance et des approches de test à mettre en œuvre). Une première approche de test, basée sur le taux d'erreur de lecture a également été proposée.

La thèse de R. Kheddam, dont la soutenance est prévue en avril 2014, a permis d'analyser la tolérance aux fautes d'un système RFID au niveau des couches logicielles (protocole et middleware). Cette thèse a abouti au développement d'un middleware RFID intégrant une approche de diagnostic probabiliste et une analyse automatique des fichiers de log des échanges entre le middleware RFID et les lecteurs RFID en vue de localiser les causes des défaillances.

Le projet DISCO

Ce projet concerne le développement d'approches de diagnostic de composants logiciels embarqués en utilisant les tests en ligne inter-composants. Ce projet a été soutenu par les financements obtenus dans le cadre du Réseau d'Excellence européen EURNEX et du projet FITT de la région Rhône-Alpes. Ce projet a été le cadre du déroulement de la thèse de Doctorat de Mme T.-Q. Bui. Bien que n'étant actuellement pas financé de manière dédiée, ce projet se poursuit sur le plan scientifique, notamment en lien avec les projets SAFERFID, CEDRE et ARTeco.

2.4. Projets de recherche antérieurs

Le projet CEDRE « Sûreté de fonctionnement des réseaux de capteurs » (janvier 2011 – décembre 2012)

Il s'agit d'un projet de collaboration avec le laboratoire LASTRe de l'Université Libanaise sur le thème de la sûreté de fonctionnement des réseaux de capteurs. J'ai assuré la responsabilité de ce projet pour le laboratoire LCIS. Le financement obtenu (10 K€ pour chaque équipe) est dédié aux déplacements et séjours des chercheurs.

L'objectif scientifique de ce projet est l'étude de la sûreté de fonctionnement des réseaux de capteurs, notamment à travers la résolution des problématiques suivantes : proposition d'un modèle de fautes adapté aux réseaux de capteurs, développement de test en ligne pour les réseaux de capteurs, diagnostic global concernant le routage et la plate-forme de gestion à partir des résultats des tests en ligne et reconfiguration. Ce projet a été le cadre du déroulement de la thèse en co-tutelle de Mme D. Hamdan.

Le projet ARTeco (janvier 2010 – décembre 2011)

Ce projet est une collaboration entre le laboratoire LCIS (équipe CTSYS) et le laboratoire LIG (équipe Sigma) de Grenoble dans le cadre du BQR de Grenoble INP. Il aborde le thème de la tolérance aux fautes des applications à base de réseaux de capteurs dans l'optique de proposer des solutions qui tiennent compte de la limitation d'énergie et des flux de données traités par ce type d'applications. De par le thème abordé, ce projet est le complémentaire du projet CEDRE ci-dessus. En effet, dans le projet ARTeco, l'accent est mis principalement sur le côté applicatif des réseaux de capteurs avec la prise en compte des flux de données échangés par les applications. La subvention obtenue pour mener ces travaux est de 20 K€ pour les 2 équipes. L'équipe projet comprend deux permanents de chaque laboratoire.

Sûreté des applications ferroviaires dans le cadre du Réseau d'excellence EURNEX www.eurnex.net (janvier 2004 – janvier 2008)

J'ai été responsable de la collaboration du laboratoire LCIS avec un ensemble d'universités et d'organismes de recherche européens (INRETS en France, FAV en Allemagne, IST au Portugal, *etc.*) dans le cadre du réseau d'excellence EURNEX dont le but était de rassembler l'ensemble des recherches dans le ferroviaire au niveau européen pour en dégager des pôles d'excellence selon le secteur étudié. Le travail accompli a consisté essentiellement en une étude de l'existant par l'analyse de projets menés tant en Europe qu'en Amérique du Nord.

Cette analyse de l'existant a permis de déterminer les pôles de recherche à promouvoir[32]. Le montant alloué au laboratoire LCIS était de 20,31 K€.

Diagnostic dans les composants logiciels embarqués dans le cadre du Fonds d'Incitation au Transfert de Technologie (FITT) de la Région Rhône-Alpes (avril 2005 – décembre 2007)

J'ai été responsable de cette étude qui a porté sur le thème du *diagnostic dans les composants logiciels embarqués*. Le résultat de ce projet a été le développement d'un service de diagnostic pour les composants logiciels embarqués, se basant sur des tests inter-composants en ligne. Le montant alloué à ce projet était de 20 K€.

Diagnostic dans les réseaux embarqués utilisés dans les applications ferroviaires, en partenariat avec l'INRETS (octobre 2005 – juillet 2006)

L'objectif de ce travail de collaboration, dont j'ai assuré la responsabilité, était d'étudier la faisabilité des tests inter-composants pour améliorer la *sûreté de fonctionnement des réseaux embarqués* utilisés dans les applications ferroviaires. Le montant de cette étude était de 14,352 K€.

Robustesse des systèmes RFID, en partenariat avec l'entreprise Schneider (février 2009 – septembre 2009)

Le but de cette collaboration était l'*analyse des défauts* pouvant survenir dans un *système RFID*. Le montant alloué à cette étude était de 5 K€.

Action spécifique programmée « Unification des méthodes de test » (septembre 1998 – septembre 2000)

Cette action visait à rapprocher les équipes de recherche dans le domaine du test logiciel et du test matériel et à étudier la possibilité d'unifier les méthodes de test proposées dans chacun de ces domaines. J'ai participé à ces travaux sur les aspects liés à *l'analyse de testabilité du logiciel, le test de haut niveau du matériel et la validation de systèmes logiciels-matériels*.

Programme PROTEUS entre le laboratoire LCIS et l'Institut Jozef Stefan de Ljubljana en Slovénie (septembre 1998 - juin 1999)

J'ai participé à ce programme sur des aspects relatifs au *diagnostic de systèmes logiciels-matériels*.

2.5. Encadrement scientifique

Co-encadrement de thèses de Doctorat

- Thèse de Brahim Sahbi sur le thème de la conception conjointe matérielle/logicielle de systèmes embarqués sûrs de fonctionnement. Cette thèse se déroule dans le cadre d'une convention CIFRE avec l'entreprise Defacto Technologies à Moirans et a débuté en mars 2014.

- Thèse de Rafik Kheddam sur le thème « Approches logicielles de sûreté de fonctionnement pour les systèmes RFID ». Directeur de thèse : I. Parissis (soutenue le 9 avril 2014). Taux d'encadrement : 75%. Financement : projet ANR SAFERFID.

- Thèse de Dima Hamdan, en co-tutelle avec l'Université Libanaise, sur le thème « Détection et diagnostic des fautes dans des systèmes à base de réseaux de capteurs sans fil ». Directeur de

[32] Ce projet a abouti à la création de l'association EURNEX dont le but est la promotion de projets de recherche dans le secteur ferroviaire européen.

thèse : I. Parissis (avril 2010 – février 2013). Thèse soutenue le 20 février 2013. Taux d'encadrement : 50%. Financement : bourse de mobilité de l'AUF.

- Thèse de Gilles Fritz sur le thème « Simulation de fautes pour l'évaluation du test en ligne de systèmes RFID ». Directeur de thèse : V. Beroulle (octobre 2009 – décembre 2012). Thèse soutenue le 10 décembre 2012. Taux d'encadrement : 50% la première année, 30% les deux dernières années en raison de l'arrivée d'un 3ème co-encadrant. Financement : Allocation de Recherche du Ministère.

- Thèse de Thi-Quynh Bui sur le thème « Service de diagnostic en ligne pour les applications à base de composants logiciels » (octobre 2005 – octobre 2009) Directeur de thèse : M. Dang. Thèse soutenue le 14 octobre 2009. Taux d'encadrement : 80%. Financement : Bourse franco-vietnamienne 25% et fonds propres LCIS 75%.

Participation à l'encadrement de thèse de Doctorat

- Thèse de Maïsaa Khalil sur le thème « Stratégies de diagnostic de systèmes logiciels-matériels » (septembre 1998 – décembre 2000)
Directeur de thèse : C. Robach
Thèse soutenue en 2002

Encadrement de stages de Master Recherche (M2R et DEA) et post-doc

- S. Boughazi (50%) – M2R Informatique et Télécommunications de l'Université P. Sabatier Toulouse 3. Titre du mémoire « Conception d'une solution d'analyse des flux de données RFID en vue du diagnostic de fautes », septembre 2013.
- M.-D. Nguyen (50%) – Post-doc dans le cadre d'un contrat ATER à l'Esisar et au LCIS durant l'année 2009-10. Les travaux menés dans le cadre de ce post-doc ont concerné les spécifications d'un middleware RFID tolérant aux fautes.

- T.-Q. Bui (100%) – Post-doc dans le cadre d'un contrat ATER à l'Esisar et au LCIS durant l'année 2009-10. Les travaux menés dans le cadre de ce post-doc ont concerné l'évaluation des performances du service DISCO.

- L. D'Orazio (100%) – Post-doc dans le cadre d'un contrat ATER à l'Esisar et au LCIS durant l'année 2007-08. Les travaux menés dans le cadre de ce post-doc ont concerné l'évaluation des besoins en mémoire des tests inter-composants logiciels.

- I. Belkacem (50%) – Magister de l'Université d'Oran de décembre 2009 à juin 2011, sur le thème « Analyse des données d'un système RFID en vue de sa sûreté de fonctionnement ».

- Q. Tran Ngoc (50%) – M2R Informatique de l'Université J. Fourier Grenoble I, durant l'année 2010-11. Le thème de ce stage, soutenu en septembre 2011, concerne l'automatisation de l'intégration d'une interface de test dans les composants logiciels.

- G. Fritz (50%) – M2R Micro et Nano Technologie de Grenoble INP. Titre du mémoire « Sûreté de fonctionnement des systèmes RFID », juin 2009.

- T.-Q. Bui (100%) – M2R Informatique de l'Université J. Fourier Grenoble I. Titre du mémoire « Un service de diagnostic pour les composants logiciels embarqués », juin 2005.

- T. Abid (100%) – DEA Informatique de l'Université J. Fourier Grenoble I. Titre du mémoire « Partitionnement d'applications réparties en vue d'un diagnostic efficace », juin 2003.

- A. Khalladi (100%) – DEA Informatique de l'Université d'Orléans. Titre du mémoire « Test de composants logiciels répartis », septembre 2003.

- A. Gacemi (100%) – DEA Informatique de l'Université J. Fourier Grenoble I. Titre du mémoire « Le diagnostic dans l'architecture CORBA », juin 2002.

- T. Mammeri (50%) – DEA Informatique de l'Université J. Fourier Grenoble I. Titre du mémoire « Apport du diagnostic distribué dans l'administration de réseaux locaux », juin 2002.

Autres encadrements en lien avec la recherche

- M. Sarkis (30%), stage de fin d'études Ingénieur et Master Recherche de l'Université Libanaise. Titre du mémoire « Test et diagnostic dans les systèmes de réseaux de capteurs hétérogènes », juin 2012.
- C. Diop (50%), stage de fin d'études de l'ENSA, Oujda, Maroc. Titre du mémoire « Développement d'un middleware RFID intégrant des services de test et de diagnostic », juillet 2012.
- J.-F. Bocquet (50%), stage de fin d'études Ingénieur de l'Esisar (Grenoble INP). Titre du mémoire « Implémentation d'un service de diagnostic pour composants logiciels », juin 2007.

- K. Kosecki (50%), stagiaire Erasmus, Pologne. Titre du mémoire « Diagnosis of embedded networks », juin 2007.

- B. Michel (100%), stage d'initiation à la recherche au laboratoire LCIS. Titre du mémoire « Sûreté de fonctionnement des systèmes et des applications réparties » été 2001

- S. Baeriswyl (50%), stage d'initiation à la recherche au laboratoire LCIS. Titre du mémoire « Conception en vue du test » été 2001

- A. Chauché (100%), thèse CNAM. Titre du mémoire « Test de haut niveau des circuits programmables en vue d'une application aux systèmes mixtes logiciels-matériels», juin 1998.

- G. Bucchi (50%), stage d'initiation à la recherche de l'IUSPIM (Marseille), effectué au LCIS. Titre du mémoire « Etude du système tolérant aux fautes SIFT », été 1998

- J.-C. Joubert (50%), stage d'initiation à la recherche de l'IUSPIM (Marseille), effectué au LCIS. Titre du mémoire « Etude du système tolérant aux fautes MAFT », été 1998

- F. Arnaud (30%), projet de fin d'études de l'IUT de Valence effectué au laboratoire LCIS. Titre du mémoire « Réalisation d'un générateur de tests fonctionnels pour des systèmes logiciels-matériels ».

- F. Chevassu et B. Rémy (80%), stage de fin d'études Ingénieur de l'Ensimag (Grenoble INP). Titre du mémoire « Evaluation des performances des algorithmes de diagnostic distribué », juin 1995.

Autres activités

- Membre du Comité de Lecture de MajecSTIC 2009 (MAnifestation des JEunes Chercheurs en Sciences et Technologies de l'Information et de la Communication), Avignon, 2009.

- Séminaire de recherche invité à l'Université d'Oran sur le thème « Sûreté de fonctionnement des systèmes RFID », avril 2009, février 2012.

3. Activités d'enseignement

L'enseignement à l'Esisar vise la formation d'ingénieurs polyvalents dans les domaines des systèmes industriels avancés. Plus précisément, cet enseignement se divise actuellement en deux filières ; la première concerne une spécialisation en « Électronique, Informatique, Systèmes » (EIS) alors que la seconde forme des ingénieurs « Informatique Réseaux » (IR). J'ai eu la charge, depuis mon recrutement en septembre 1999, d'organiser un cursus de « Système » pour répondre au mieux aux demandes de la filière EIS, puis des deux filières EIS et IR. Un objectif important concernait l'introduction du « Temps réel », qui est un enseignement clé à l'Esisar. J'ai également été en charge de développer un enseignement de « Sûreté de fonctionnement des systèmes informatiques ».

3.1. Système

L'enseignement de système à l'Esisar a pour objectif de fournir aux futurs ingénieurs les outils de base pour l'utilisation, la programmation et la conception de systèmes d'exploitation modernes. Cet enseignement s'articule autour de 2 thèmes détaillés ci-dessous.

Système d'exploitation

Dans ce thème, l'accent est essentiellement mis sur deux points :

- l'utilisation d'un système d'exploitation ; qui consiste à permettre aux élèves de se familiariser avec les commandes de base d'un système (1ère année Esisar), à l'écriture de scripts shell (2ème année Esisar) mais également à utiliser des fonctions plus avancées de création de processus et de communication inter-processus (3ème année Esisar).
- la conception d'un système d'exploitation ; il s'agit ici de présenter les différentes fonctions d'un système d'exploitation sous un angle plus théorique. Ainsi sont étudiés, entre autres thèmes, les principaux algorithmes d'ordonnancement, la gestion de la mémoire, la mémoire virtuelle, le problème de l'interblocage. Cet enseignement se déroule en 3ème année de l'Esisar.

Systèmes et applications répartis[33]

Cet enseignement concerne les modèles utilisés pour le développement de systèmes et applications répartis. Les points étudiés sont :

- le modèle client-serveur objet à travers l'appel de méthode à distance de Java,
- le modèle à agents mobiles,
- le modèle à objets dupliqués et l'étude des protocoles de gestion de la cohérence,
- le modèle CORBA et son apport dans le développement d'une norme pour la construction d'applications réparties,
- le modèle à composants (OSGi et composants CORBA notamment).

3.2. Temps réel

L'Esisar vise la formation d'ingénieurs compétents dans le domaine des systèmes embarqués, pour lesquels le temps réel est de toute première importance. L'enseignement du

[33] Ce cours a été assuré la première fois en 1999-2000 par M. Daniel Hagimont, alors Chercheur dans le projet Sardes de l'INRIA. J'ai par la suite repris les principaux thèmes de ce cours.

temps réel à l'Esisar est un enseignement transversal dans le sens où il relève de trois départements : Électronique, Informatique et Automatique.

J'ai pour ma part été responsable de la définition et de la mise en place des enseignements qui relèvent du département Informatique et qui concernent la conception et la mise en œuvre de systèmes temps réel, la programmation réactive ainsi que des enseignements sur la structure et le fonctionnement des noyaux temps réel.

Méthodes de programmation réactive

Ce cours a présenté deux principales méthodes de modélisation et de développement des systèmes temps réel : l'approche *synchrone*, illustrée par le langage LUSTRE, et l'approche *asynchrone* avec le langage LOTOS. En raison de mon implication dans des charges collectives relativement importantes, ce cours a été délégué à un vacataire extérieur à partir de l'année 2002-2003.

Noyaux temps réel

J'ai eu la responsabilité de la mise en place de ce cours qui se focalise sur la présentation des principales fonctions d'un noyau temps réel et les différentes techniques d'ordonnancement qui réalisent le respect des échéances temporelles. J'ai également assuré la mise en place de deux versions pour les TP. La première version a concerné la programmation d'un noyau temps réel réduit. Elle utilisait l'environnement de développement Micrototec de Mentor Graphics sur des cibles de type Motorola 68020. Par la suite, des échanges que j'ai eus avec quelques industriels (ALCATEL, MONETEL et ASCOREL en particulier) ont motivé l'orientation de ces TP vers le développement d'applications temps réel à partir d'un noyau existant. Ainsi, depuis l'année 2002-2003, ces TP ont concerné le développement de programmes faisant appel à des tâches temps réel sur une plate-forme VRTX de Mentor Graphics. Les cibles matérielles utilisées sont des cartes MVME 162 à microprocesseur Motorola 68040. Pour la mise en place de ces deux versions de TP, une forte collaboration a eu lieu avec l'enseignant de Microprocesseurs afin de permettre aux élèves de bien situer les différents composants d'un système embarqué temps réel.

3.3. Sûreté de fonctionnement

Cet enseignement, dont j'ai assuré la mise en place, présente les concepts de base de la sûreté de fonctionnement et de la tolérance aux fautes dans les systèmes informatiques, ainsi que les principales techniques d'évaluation de la sûreté de fonctionnement. Il s'inscrit dans le cadre d'un module général de qualité enseigné en dernière année de l'Esisar.

3.4. Autres enseignements

Bien que mes enseignements principaux concernent les systèmes d'exploitation, le temps réel et la sûreté de fonctionnement, je me suis également impliquée dans d'autres types d'enseignement. Ainsi, j'ai également assuré des enseignements d'algorithmique, de langage C et de programmation Java à l'Esisar ainsi que des enseignements d'architecture des microprocesseurs, de composants logiciels et de réseaux au département d'Informatique de l'IUT 2 de Grenoble, dans le cadre de vacations. J'ai également assuré pendant plusieurs années des cours de système Unix auprès des étudiants de l'IUT de Valence, dans le cadre de vacations.

3.5. Activités et encadrements de type pédagogique

Collaboration avec l'Université d'Oran

Je collabore depuis avril 2009 à la post-graduation « Ingénierie des Données et des Connaissances » de l'Université d'Oran en Algérie pour laquelle j'assure un cours de sûreté de fonctionnement et de sécurité. Cette formation a lieu sur 2 journées et est couplée avec un séminaire recherche que j'anime. La première session de formation/séminaire a permis de définir des axes de collaboration en recherche qui ont permis le co-encadrement d'un stage de Magister portant sur le thème de l'analyse des données d'un système RFID en vue de sa sûreté de fonctionnement (soutenu en juin 2011).

Projet BQE (avril 2005 – avril 2007)

J'ai assuré la responsabilité d'un projet BQE (Bonus Qualité Enseignement) de Grenoble INP qui fédère une équipe de 5 permanents de l'Esisar (1 Professeur des Universités en automatique, 1 Maître de Conférences en informatique, 2 Ingénieurs-Enseignants en électronique et 1 Ingénieur-Système) dans l'objectif d'une *restructuration de l'enseignement de temps réel* à l'Esisar en vue de mieux répondre aux nouvelles problématiques qui se posent (modélisation objet des systèmes temps réel, lien entre l'informatique, l'automatique et l'électronique dans un système temps réel, *etc*.). La subvention allouée à ce projet était de 15,8 K€.

Collaboration industrielle sur les noyaux temps réel (juin 2002)

Cette collaboration avait pour objectif d'aider l'entreprise ASCOREL, située à Pont-Evêque dans l'Isère, à introduire les noyaux temps réel dans les systèmes de commande automatique de grues qu'elle conçoit. Cette collaboration s'est déroulée sous forme de *deux journées de formation* du personnel de l'entreprise aux noyaux temps réel et d'*une journée de conseil* durant laquelle le problème spécifique d'ASCOREL a été étudié et analysé.

Collaboration entre l'Esisar et la société Électronique Générale EGVA (1998-1999)

Il s'agit d'un contrat de collaboration entre l'Esisar et la société Électronique Générale EGVA, dont le siège social se trouve à Montmélian, Drôme. Cette société étudie et réalise des produits de sécurité pour les remontées mécaniques utilisant des ensembles électroniques. Elle vise à moderniser ces équipements par le biais de la microélectronique, tout en tenant compte de la sécurisation des logiciels embarqués dans les microcontrôleurs. Dans le cadre de ce contrat, j'ai été chargée d'assurer une formation aux méthodes de sûreté de fonctionnement pour le personnel de la société.

Encadrement pédagogique de projets industriels de l'Esisar

Je m'implique de manière régulière dans l'encadrement pédagogique de projets industriels de l'Esisar (la charge liée à cet encadrement équivaut à 80 heures équivalent TD). Les projets que j'ai encadrés sont les suivants :

- C. Chardon, X. Dessemon et L. Rabeuf, janvier-juillet 2014. Entreprise Trixell, Moirans ; sur le thème « Solution d'algorithmes de correction d'image sur un matériel portable à Rayon X ».

- A. Joudar, S. Monticelli et K. Pagès, janvier-juillet 2010. Entreprise Thalès, Valence ; sur le thème «Banc de prototypage rapide d'algorithmes de navigation inertielle hybride».

- L. Giron-dit-Metaz, L. Minaudier, N. Tritz, janvier-juillet 2009. Entreprise Thalès, Valence ; sur le thème «Portage du logiciel de Flight Management System (FMS) sur OS temps réel».

- J. Bonnard, A. Flandin et R. Rebotier, janvier-juillet 2008. Entreprise ST Microelectronics, Crolles ; sur le thème «Développement de couches logicielles spécifiques : SAMPLING AEMP».

- V. Bardol, M. Bienvenue, H. Rifflet, janvier-juillet 2007. Entreprise Ascorel, Pont-Evêque ; sur le thème «Module de télémaintenance via GPRS 3G pour système de pesage embarqué».

- V. Delhommois, F. Dutron, G. Kalaszczaj, janvier-juillet 1999. Entreprise Krohne, Romans-sur-Isère ; sur le thème « Système de télémaintenance via Intranet pour détecteurs de niveau».

Tutorat et implications diverses (depuis l'année 1999)

J'assure des activités régulières de tutorat de projets de fin d'études (3 à 4 par an) et de projets de techniciens (2 à 3 par an).

Depuis la mise en place de la voie par apprentissage de la filière EIS à l'Esisar, j'assure régulièrement le tutorat, sur 3 années, d'un élève-apprenti (A. Pigeaut avec l'entreprise Imaje de 2006 à 2009, S. Atak avec l'entreprise Thalès de 2008 à 2011, R. Dethurens avec l'entreprise Thalès de 2010 à 2013, T. Robert et M. Sezzano avec l'entreprise Sagem depuis la rentrée 2012). Ce type de tutorat implique un accompagnement sur la durée des élèves-apprentis pour les amener, conjointement avec le Maître de stage en entreprise, à acquérir les compétences et aptitudes d'un ingénieur.

Pour compléter les enseignements de 5ème année à l'Esisar, j'invite régulièrement des spécialistes (temps réel, systèmes multi-agents, middleware RFID, *etc*.) pour des conférences dans leurs domaines respectifs.

Enfin, je participe régulièrement aux différents jurys d'admission à l'Esisar (post bac et bac+2) ainsi qu'à diverses manifestation et salons pour promouvoir l'image de l'Esisar.

4. Charges collectives et d'intérêt général

Ayant intégré l'Esisar et le LCIS, tous deux jeune école d'ingénieurs et jeune laboratoire de recherche, je me suis très tôt impliquée dans des actions collectives et d'intérêt général tant au niveau de l'enseignement que de la recherche.

4.1. Charges actuelles à l'Esisar

- Responsable de la 5ème année de l'Esisar depuis septembre 2007.

4.2. Charges antérieures à l'Esisar et Grenoble INP

- Membre nommé du Conseil du laboratoire LCIS de septembre 2007 à août 2011.

- Pour l'année 2010-11, j'ai été membre du Comité de Sélection d'un poste de Maître de Conférences, section 27, à l'Esisar.

- Responsable du département informatique de l'Esisar de septembre 2007 à août 2009.

- Responsable de l'équipe de recherche CSYS (Conception et sûreté de fonctionnement des SYStèmes distribués) au laboratoire LCIS de septembre 2003 à septembre 2007 (cette équipe comprenait 3 permanents et 2 doctorants).

- Membre élu du Conseil de l'Esisar de septembre 2000 à juillet 2008.

- Membre extérieur nommé de la Commission de Spécialistes Informatique de l'Université Joseph Fourier de septembre 2000 à juillet 2006.

- Chargé de Mission auprès de la Mission Pédagogique de l'Institut Polytechnique de Grenoble de mars 2001 à juillet 2006, sous la responsabilité du Vice-Président CEVU.

- Responsable de la 3ème année à l'école d'Ingénieurs Esisar de septembre 2002 à août 2005.

- Membre élu de la Commission de Spécialistes Informatique de l'Institut Polytechnique de Grenoble de septembre 2001 à août 2003.

- Responsable déléguée de la 5ème année Esisar de septembre 1997 à août 1999.

4.3. Charges antérieures dans d'autres établissements

- Responsable de l'enseignement d'Informatique en DEUG Économie et Gestion, Université Grenoble II (de septembre 1996 à août 1997) durant ma première année d'ATER.

- Organisation de séminaires de recherche (de janvier à juin 1995) au sein du groupe de recherche GRAM (Groupe de Recherche en Architecture de Machines) du laboratoire LGI-IMAG à raison d'un séminaire tous les 15 jours.

5. Publications

Revues internationales avec comité de lecture

1. « SafeRFID-MW: Safe and Fault-Tolerant RFID Middleware », R. Kheddam, ***O. Aktouf***, I. Parissis, *Journal of Communications Software and Systems (JCOMMS), Special issue on RFID Technologies and Internet of Things*, Vol. 9, n° 1, mars 2013, pp. 57-73.

2. « RFID System On-line Testing based on the evaluation of the Tags Read-Error-Rate », G. Fritz, V. Beroulle, ***O. Aktouf***, D. Hély, *Journal of Electronic Testing (JETTA), Springer Netherland, vol. 27*, juin 2011, pp.267-276. DOI: 10.1007/s10839-010-5191-6.

3. « On-line Testing of Software Components for Diagnosis of Embedded Systems », T.Q. Bui et ***O. Aktouf,*** *International Journal of Computer Systems Science and Engineering*, Volume 4, number 2 (2007), pp 96-102 (sélectionné suite à la conference *The 4th International Conference on Embedded and Real-Time Computing Systems*, Prague, République Tchèque**,** juillet 2007).

4. « Host-Diagnosis Algorithms for Parallel Systems », ***O. Aktouf***, *Journal of Systems Architecture* 47 (2001), pp. 675-695.

5. « Performance Evaluation of Distributed Diagnosis Algorithms in Parallel Systems », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *Journal of Parallel Computing*, 24 (1998), pp. 1205-1222.

Revues nationales avec comité de lecture

6. « Diagnostic des architectures parallèles : un état de l'art », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *Technique et Science Informatiques*, vol. 16, n°2, février 1997, pp. 195-224.

Chapitre d'ouvrage

7. « Basic Concepts of Fault-Tolerant Computing Design », C. Aktouf et ***O. Aktouf-Benkahla***, Chapitre 11 dans « *Uncertainty Modeling in Finite Element, Fatigue and Stability of Structures* », *Series on Stability, Vibrations and Control of Systems*, vol. 9, *World Scientific Publishing* 1997, pp. 357-394.

Conférences et colloques internationaux avec actes et comité de lecture, sélection sur papier complet

8. « Monitoring of RFID Failures Resulting from LLRP Misconfigurations », R. Kheddam, ***O. Aktouf***, I. Parissis et S. Boughazi, *21th IEEE International Conference on Software, Telecommunications and Computer Networks* (*SOFTCOM 2013*), Split, Croatie, septembre 2013, pp. 1-6.

9. « Online data fault detection in wireless sensor networks », M. Sarkis, D. Hamdan, B. El Hassan, ***O. Aktouf*** et I. Parissis, *IEEE Second International Conference on Advances in Computational Tools for Engineering Applications*, décembre 2012, Zouk Mosbeh, Liban, pp. 61-65.

10 « A Self monitoring, Adaptative and Resource Efficient Approach for Improving QoS in Wireless Sensor Networks », D. Hamdan**, *O. Aktouf***, I. Parissis, B. El Hassan, A. Hijazi et B. Moslem, *IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (Cyber'C 2012)*, 10-12 octobre 2012, Sanya, Chine, pp. 393-400.

11. « Online monitoring and diagnosis of RFID readers and tags », R. Kheddam, ***O. Aktouf*** et I. Parissis, *20th IEEE International Conference on Software, Telecommunications and Computer Networks* (*softCOM 2012*), Split, Croatie, 11-13 septembre 2012, pp. 1-9.

12. « SMART Service for Fault-Diagnosis in Wireless Sensor Networks », D. Hamdan**, *O. Aktouf***, I. Parissis, A. Hijazi, M. Sarkis et B. El Hassan, *IEEE 6th International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2012)*, 12-14 septembre 2012, Paris, France, pp. 211 – 216.

13. « Online Data Fault Detection for Wireless Sensor Networks - Case Study », D. Hamdan**, *O. Aktouf***, I. Parissis, B. El Hassan et A. Hijazi, *IEEE Third International Conference on Wireless Communication in Unusual and Confined Areas (ICWCUCA 2012)*, 28-30 août 2012, Clermont-Ferrand, France, pp. 1-6.

14. « An extended LLRP model for RFID system test and diagnosis », R. Kheddam, ***O. Aktouf*** et I. Parissis, *8th Workshop on Advances in Model Based Testing (A-MOST 2012)* dans le cadre de *5th IEEE International Conference on Software Testing, Verification and Validation*. Montreal, Canada, 17-21 avril 2012, pp. 529-538.

15. « Integrated Fault tolerance Framework for Wireless sensor networks », D. Hamdan**, *O. Aktouf***, I. Parissis, B. El Hassan et A. Hijazi, *IEEE 19th International Conference on Telecommunications*, 23-25 avril 2012, Jounieh, Liban, pp. 1-6.

16. « Evaluation of a new RFID system performance monitoring approach », G. Fritz, V. Beroulle, ***O. Aktouf*** et D. Hély, *Design, Automation & Test in Europe Conference & Exhibition (DATE 2012)*, 12-16 mars 2012, pp.1439-1442.

17. « Read rate profile monitoring for defect detection in RFID Systems », G. Fritz, B. Maaloul, V. Beroulle, ***O. Aktouf***, D. Hély, *2011 IEEE International Conference on RFID - Technology and Applications (RFID-TA 2011)*, 15-16 septembre 2011, Sitges, Espagne, pp. 89-94.

18. « SystemC Modeling of RFID Systems for Robustness Analysis », G. Fritz, V. Beroulle, ***O. Aktouf***, D. Hély, *IEEE 19th International Conference on Software Telecommunications and Computer Networks (softCOM) – Symposium on RFID Technology and Applications*, 15-16 septembre 2011, pp. 1-5.

19. « Towards middleware-based fault-tolerance in RFID systems », M.D. Nguyen, G. Fritz, ***O. Aktouf***, V. Beroulle et D. Hély, *13th European Workshop on Dependable Computing (EWDC 2011)*, Pise, Italie, 11-12 mai 2011, pp. 49-52.

20. « Analyse des données d'un système RFID en vue de sa sûreté de fonctionnement », I. Belkacem, ***O. Aktouf*** et S. Nait Bahloul, *1st International Conference on Information Systems and Technologies (ICIST 2011)*, 24-26 avril 2011, Tébessa, Algérie.

21. « Vers la tolérance aux fautes dans les systèmes RFID », I. Belkacem, ***O. Aktouf*** et S. Nait Bahloul, *International Conference on Systems and Processing Information (ICSPI 2011)*, 15-17 Mai 2011, Guelma, Algérie.

22. « Read-error-rate evaluation for RFID system on-line testing », G. Fritz, V. Beroulle, M.D. Nguyen, ***O. Aktouf*** et I. Parissis, *IEEE 16th International Mixed-Signals, Sensors and Systems Test Workshop*, La Grande Motte, France, 7-9 juin 2010, pp. 1-6.

23. « Dependability of Embedded Networks – A Case Study with System Diagnosis of CAN Protocol », ***O. Aktouf***, Ch. Deleuze et M. Wahl, *9th International Conference on ITS Telecommunication*, Lille, France, 20-22 octobre 2009, pp. 552-556.

24. « Diagnosis Service for Software Components and its Application to a Heterogeneous Sensor Data Management System », T.Q. Bui, ***O. Aktouf***, M. Dang, L. Gürgen et C. Ronconcio, *Second International Conference on Dependability, Depend 2009*, 18-23 juin 2009, Athènes, Grèce, pp. 143-149.

25. « Service de diagnostic pour composants logiciels », T.Q. Bui, ***O. Aktouf***, L. D'Orazio et M. Dang, *Qualita 2009, Besançon, France*, 18-20 mars 2009, pp. 1-9.

26. « Software Component Diagnosis Service: Architecture Description », T.Q. Bui, ***O. Aktouf*** et M. Dang, *Third IEEE International Symposium on Industrial Embedded Systems, La Grande Motte, France*, 11-13 juin 2008, pp. 134-140.

27. « Diagnosis Service for Embedded Software Component based Systems », T.Q. Bui et ***O. Aktouf***, *The 2nd International Workshop on Engineering Fault Tolerant Systems (EFTS 2007)*, Dubrovnik, Croatie, 4 septembre 2007**,** pp. 14-19.

28. « On-line Testing of Software Components for Diagnosis of Embedded Systems », T.Q. Bui et ***O. Aktouf,*** *The 4th International Conference on Embedded and Real-Time Computing Systems, Prague, République Tchèque***,** juillet 2007, pp.330-336.

29. « Inter-component testing for system-level diagnosis of embedded component based applications », T.Q. Bui et ***O. Aktouf,*** *The 7th Multidisciplinary International Conference on Quality and Reliability*, Tanger, Maroc, 20-22 mars 2007, pp. 246-254.

30. « Component Diagnosis for Distributed Applications », ***O. Aktouf*** et A. Gacemi, *Actes du Congrès Qualita*, Nancy, France, 18-20 mars 2003, pp. 129-136.

31. « Periodic Testing of FPGAs Using a Software Test Method », ***O. Aktouf-Benkahla***, A. Chauché, C. Aktouf et C. Robach, *5th IEEE International On-Line Testing Workshop, Rhodes, Grèce*, 5-7 juillet 1999.

32. « Testing Programmable Devices in Hardware/Software Systems », ***O. Aktouf-Benkahla***, A. Chauché, C. Aktouf et C. Robach, *IEEE European Test Workshop (ETW'99)*, 25-28 mai 1999, Constance, Allemagne.

33. « Off-line Diagnosis of Parallel Systems », ***O. Aktouf-Benkahla*** et C. Robach, *17th IEEE Symposium on Reliable Distributed Systems (SRDS'98)*, 20-22 octobre 1998, West Lafayette, USA, pp. 360-365.

34. « Performance Evaluation of Testing Strategies in Parallel Systems », ***O. Aktouf-Benkahla***, F. Chevassu, B. Rémy et C. Robach, *22nd IEEE Euromicro Conference*, 2-5 septembre 1996, Prague, République Tchèque, pp. 371-378.

35. « Distributed System-Level Diagnosis of Parallel Architectures », ***O. Aktouf-Benkahla***, C. Aktouf, F. Chevassu, B. Rémy et C. Robach, *2nd IEEE International Conference on Massively Parallel Computing Systems (MPCS'96)*, 6-9 mai 1996, Ischia, Italie, pp. 238-245.

36. « System-Diagnosis of Cluster-Based Parallel Architectures », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *IEEE Euromicro Workshop on Parallel and Distributed Processing*, 24-26 janvier 1996, Braga, Portugal, pp. 305-309.

37. « Distributed Off-Line Testing of Parallel Systems », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *The IEEE Fourth Asian Test Symposium (ATS'95)*, 23-24 novembre 1995, Bangalore, Inde, pp. 2-8.

38. « Host-Diagnosis of Parallel Systems », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'95)*, CSREA Press, 3-4 novembre 1995, Géorgie, USA, pp. 148-157.

39. « A Diagnosis Strategy for Cluster-Based Parallel Architectures », ***O. Aktouf-Benkahla***, C. Aktouf et C. Robach, *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'95)*, CSREA Press, 3-4 novembre 1995, Géorgie, USA, pp. 111-120.

40. « Functional Testing of Parallel Message-Passing Machines », C. Aktouf, ***O. Aktouf-Benkahla*** et C. Robach, *21st IEEE Euromicro Conference*, 4-6 septembre 1995, Côme, Italie, pp. 413-420.

41. « Distributed Validation of Massively Parallel Machines », C. Aktouf, ***O. Aktouf-Benkahla*** et C. Robach, *International IEEE Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'94)*, 14-16 décembre 1994, Kanazawa, Japon, pp. 326-333.

Conférences et colloques internationaux/nationaux avec actes et comité de lecture, sélection sur résumé

42. « Runtime quality services assurance for Wireless sensor and actuator networks », D. Hamdan, ***O. Aktouf***, I. Parissis, A. Hijazi et B. El Hassan, *18th International Science Meeting*, 22-24 mars 2012, Zouk Mikael, Liban.

43. « Introducing fault-diagnosis into embedded CORBA-based systems », ***O. Aktouf***, M. Wahl et M. Dang, *IEEE International Conference on Information and Communication Technologies: from Theory to Applications (ICTTA) 2004*, Syrie, 19-23 avril 2004, pp 679-680.

44. « A Diagnosis Service for CORBA-Based Applications », ***O. Aktouf*** et A. Gacemi, *Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE) 2002*, Annapolis, Maryland, USA, 12-15 novembre 2002, pp. 185-186.

45. « An Experimental Comparison of Software Diagnosis Methods », M. Khalil, ***O. Aktouf-Benkahla*** et C. Robach, *Euromicro 2nd Workshop on Dependable Computing Systems, IEEE Euromicro Conference*, Milan, Italie, 8-10 septembre 1999, pp. 146-149.

46. « Performance Comparison of Self-Diagnosis Algorithms », ***O. Aktouf-Benkahla***, F. Chevassu, B. Rémy et C. Robach, *2nd IEEE International On-Line Testing Workshop*, 8-10 juillet 1996, Biarritz, France, pp. 187-191.

47. « Hierarchical Distributed Diagnosis of Parallel Message-Passing Machines », C. Aktouf, ***O. Aktouf-Benkahla*** et C. Robach, *1st IEEE International On-Line Testing Workshop*, 4-5 juillet 1995, Nice, France, pp. 37-41.

48. « On Distributed Diagnosis of Parallel Systems », C. Aktouf, ***O. Aktouf-Benkahla*** et C. Robach, *SIPAR Workshop on Parallel and Distributed Computing*, 14 Octobre 1994, Fribourg, Suisse, pp. 73- 77.

Conférences et colloques nationaux avec actes et comité de lecture

49. « Algorithme de diagnostic probabiliste pour la détection des défaillances dans les systèmes RFID », R. Kheddam, ***O. Aktouf*** et I. Parissis, *9ème Conférence Internationale Jeunes Chercheurs (MajecSTIC 2012)*, Lille, France, 29-31 octobre 2012, pp. 1-8.

50. « Détection des fautes dans les réseaux des capteurs sans fils », M. Sarkis, D. Hamdan, ***O. Aktouf*** et I. Parissis, Poster présenté à *9ème Conférence Internationale Jeunes Chercheurs (MajecSTIC 2012)*, Lille, France, 29-31 octobre 2012.

51. « Sûreté de fonctionnement des systèmes RFID – Evaluation et simulation des dysfonctionnements », G. Fritz, V. Beroulle, ***O. Aktouf*** et M. D. Nguyen, *Journées Nationales du Réseau Doctoral en Microélectronique*, Montpellier, France, 7-9 juin 2010.

Papiers invités

52. « Inter-component testing for system-level diagnosis of CORBA-based applications », ***O. Aktouf***, T.-Q. Bui et M. Dang, *Proceedings of EURNEX Pole 5 « Intelligent Mobility » Scientific Workshop*, Lille, France, octobre 2005.

53. « Hardware/Software Design and Test: Challenges and Opportunities », I. Teixeira, J.-P. Teixeira, ***O. Aktouf-Benkahla*** et C. Robach, *IEEE International Conference on Dynamics and Control of Systems (DYCONS'99)*, 5-7 août 1999, Ottawa, Canada.

Autres

54. « Sûreté de fonctionnement des réseaux embarqués », ***O. Aktouf*** et Ch. Deleuze, *Rapport d'étude INRETS, juillet 2006.*

55. *« Détection et localisation de composants embarqués défaillants », rapport Projet FITT, Juin 2006.*

56. « Diagnostic des Architectures Parallèles à Passage de Messages », *Thèse de Doctorat d'Informatique de l'INP Grenoble*, Juin 1997.

57. « Diffusion sous Contraintes Temporelles dans les Architectures Massivement Parallèles », *Mémoire de DEA d'Informatique de l'INP Grenoble*, Juin 1993.

Annexe B : articles joints

En lien avec le chapitre 1

Référence [67] dans le mémoire

« Diagnosis Service for Software Components and its Application to a Heterogeneous Sensor Data Management System », T.Q. Bui, O. Aktouf, M. Dang, L. Gürgen et C. Ronconcio, *Second International Conference on Dependability, Depend 2009*, 18-23 juin 2009, Athènes, Grèce, pp. 143-149.

En lien avec le chapitre 2

Référence [78] dans le mémoire

« SystemC Modeling of RFID Systems for Robustness Analysis », G. Fritz, V. Beroulle, O. Aktouf, D. Hély, IEEE 19th International Conference on Software Telecommunications and *Computer Networks (softCOM) – Symposium on RFID Technology and Applications*, 15-16 septembre 2011, pp. 1-5.

Référence [109] dans le mémoire

« Evaluation of a new RFID system performance monitoring approach », G. Fritz, V. Beroulle, O. Aktouf et D. Hély, *Design, Automation & Test in Europe Conference & Exhibition (DATE 2012)*, 12-16 mars 2012, pp.1439-1442.

Référence [130] dans le mémoire

« An extended LLRP model for RFID system test and diagnosis », R. Kheddam, O. Aktouf et I. Parissis, *8th Workshop on Advances in Model Based Testing (A-MOST 2012)* dans le cadre de *5th IEEE International Conference on Software Testing, Verification and Validation*. Montreal, Canada, 17-21 avril 2012, pp. 529-538.

Référence [79] dans le mémoire

« SafeRFID-MW: Safe and Fault-Tolerant RFID Middleware », R. Kheddam, O. Aktouf, I. Parissis, *Journal of Communications Software and Systems (JCOMMS), Special issue on RFID Technologies and Internet of Things*, Vol. 9, n° 1, mars 2013, pp. 57-73.